UNIVERSITY OF SOUTHAMPTON

Beyond multi-class – structured learning for machine translation

by

Yizhao Ni

A thesis submitted in partial fulfillment for the
degree of Doctor of Philosophy

in the
Faculty of Engineering, Science and Mathematics
School of Electronics and Computer Science

June 2010

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING, SCIENCE AND MATHEMATICS
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

Doctor of Philosophy

by Yizhao Ni

In this thesis, we explore and present machine learning (ML) approaches to a particularly challenging research area – machine translation (MT). The study aims at replacing or developing each component in the MT system with an appropriate discriminative model, where the ultimate goal is to create a powerful MT system with cutting-edge ML techniques.

The study regards each sub-problem encountered in the MT field as a classification or regression problem. To model specific mappings in MT tasks, the modern machine learning paradigm known as "structured learning" is pursued. This approach goes beyond classic multiclass pattern classification and explicitly models certain dependencies in the target domain.

Different algorithmic variants are then proposed for constructing the ML-based MT systems: the first application is a kernel-based MT system, that projects both input and output into a very high-dimensional linguistic feature space and makes use of the maximum margin regression (MMR) technique to learn the relations between input and output. It is amongst the first MT systems that work with pure ML techniques. The second application is the proposal of a max-margin structure (MMS) approach to phrase translation probability modelling in an MT system. The architecture of this approach is shown to capture structural aspects of the problem domains, leading to demonstrable performance improvements on machine translation. Finally the thesis describes the development of a phrase reordering model for machine translation, where we have compared different ML methods and discovered a particularly efficient paradigm to solve this problem.

Contents

List of Figures

List of Tables

Acknowledgements

The process of obtaining a PhD degree is not a journey that can be completed alone. I would like to thank the many individuals, named and unnamed, who have helped me throughout this journey. Particularly thanks are owing to my supervisors Dr. Craig Saunders and Prof. Mahesan Niranjan, who guided and advised me through this circuitous process. I would also like to thank Dr. Sandor Szedmak, who is always ready to share with me his vast knowledge and infinite patience.

In addition, I would like to thank Prof. Steve R. Gunn from the ISIS research group at the University of Southampton for acting as my internal PhD examiner, as well as thanking Prof. Nello Cristianini of the Pattern Analysis and Intelligent Systems research group at the University of Bristol, who was my external PhD examiner.

The following researchers with whom I have had the pleasure of meeting and exchanging ideas, Prof. John Shaw-Taylor, Prof. Juho Rousu, Assistant Prof. Philipp Koehn, Dr. Zhuoran Wang, Dr. Matti Kääriäinen, Dr. Hieu Hoang. I appreciate all your contribution through discussion and suggestions.

I would like to acknowledge the financial support of the PASCAL Network and the European Commission under the IST Project SMART (FP6-033917).

And last, but not least, I would like to thank my parents Shichuan Ni and Hong Huang. I appreciate your love and support during these years, thank you! This thesis is also the fruit of your hard work.

Chapter 1

Machine translation and machine learning

1.1 Translation

Translation is a process of interpreting the meaning of a text and subsequently producing an equivalent text, that aims at communicating the same message in another language. The text to be translated is called the *source* text, and the language that is to be translated into is called the *target* language.

The art of translation is as old as written literature (Cohen, 1986). However, it did not develop rapidly until the industrial revolution, where the growth in technology and business and the demand in communication contributed to the progress of translation. In particular, the advent of the internet has greatly expanded the market for translation and developed a wide range of requirements, such as product localisation and multilingual documentation.

Human translations are extensive but expensive. Although human translators evaluate many aspects of translation including fidelity, fluency and aesthetics, they can take weeks or even months to finish. This quickly became a big barrier to many translation businesses like interpreters. The demand for efficient and economical "translators" becomes the primary task.

In the 1950s first attempts have been made to automate the translation of natural-language texts with computers. This is the early form of *machine translation* (MT), that opened a new era of translation.

1.2 Learning how to translate

The beginning of the research in machine translation was a great success in terms of translation speed. However, it very soon became apparent that automatically translating any document at a quality equivalent to the best human translator was not only over-ambitious but itself unrealistic (Slocum, 1985). This is because of various reasons, from how to teach a computer identifying, understanding and conveying the full knowledge of morphology, syntax and semantics of both languages, to computational complexity due to a considerably large exploration space the MT problems exist. A pragmatic theory on learning how to translate is then at the top of the agenda.

Machine learning (ML) is now established as a powerful way of formulating and learning the relations between two study objects in artificial intelligence. It has significant impact to various sectors of our world, including computer vision, finance and markets, natural language processing, and so forth. Teaching a computer how to learn is the core of machine learning, which at the same time is an urgent requirement of machine translation. This fact brings about the cooperation between machine translation and machine learning and motivates us to achieve it. In this thesis, the exploration of the proposed ML approaches reveals our attempt for perfecting the theory on learning how to translate.

1.3 Thesis outline & Contributions

The outline of this thesis is divided into two main parts. Part I provides a systematic review of both machine translation and machine learning, and in Part II, three proposed models for machine translation are presented.

Part I

- Chapter 2 gives an introductory background of machine translation, that briefly summarises the history of machine translation and elaborates the framework of an MT system.
- Chapter 3 reviews several cutting-edge machine learning technologies, which are the fundamental building blocks of the proposed work throughout the thesis.

Part II

- Chapter 4 demonstrates a kernel–based MT system and the preliminary translation results on two French-to-English MT tasks.

- Chapter 5 concludes the application of an efficient structured learning approach to the *phrase translation probability* (PTP) model in an MT system, where we show the efficiency of the approach and reasonable improvements over the baseline models.
- In Chapter 6, a phrase reordering model is constructed for the state-of-the-art *statistical machine translation* (SMT) system (MOSES). The model is then evaluated by a batch of translation tasks, verifying its effectiveness in an MT system.

Finally in Chapter 7 we conclude the works presented throughout the thesis and discuss the open issues and future work.

The substantial contribution of this thesis is the investigation of a variety of machine learning methodologies for MT and the discovery of when and where to apply what technologies. Below we specify the publications and contribution that the thesis has devoted in full.

- Associated publications to chapter 5 – (Ni et al., 2009b) and (Ni et al., 2010b).
- Associated publications to chapter 6 – (Ni et al., 2009a), (Ni et al., 2010c) and (Ni et al., 2010a).
- Associated contribution to the SMART project[1] – a contribution to the deliverable "Characterisation of current and Markov based approaches" (Deliverable 2.1).
- Software contribution – a small project funded by PASCAL[2] that aims at integrating the proposed work (Chapter 6) into the public domain SMT system (MOSES[3]) is now complete. The software package is available at: `http://eprints.ecs.soton.ac.uk/20939/`.
- Corpus resource contribution – the *parallel texts of Hong Kong laws* (Chinese–English) corpus has been further revised and aligned at the sentence level by the author. This refined corpus has been submitted to Linguistic Data Consortium (LDC) for publication.

[1] `http://www.smart-project.eu/node/1`
[2] `http://www.pascal-network.org`
[3] `http://www.statmt.org/moses/`

Chapter 2

A brief history of machine translation

Symbol	Notation
$\mathbf{f}/\mathbf{e}$	a source/target sentence (string)
$\mathbf{f}^i/\mathbf{e}^i$	the i-th source/target sentences in the data set
f_i/e_i	the i-th word in the source/target sentence
N	the size of the source (target) set
$\mathbb{F}$	the input (source string) space
$\mathbb{E}$	the output (target string) space
$\bar{\mathbf{f}}^I$	the source phrase sequence
$\bar{\mathbf{e}}^I$	the target phrase sequence
$\bar{f}_i$	the source phrase where $\bar{f}$ denotes the sequence of words and i denotes $\bar{f}$ is the i-th phrase in the source phrase sequence $\bar{\mathbf{f}}^I$
$\bar{e}_{j_i}$	the target phrase where $\bar{e}$ denotes the sequence of words and j_i denotes $\bar{e}$ is the j_i-th phrase in the target phrase sequence $\bar{\mathbf{e}}^I$
$\Omega_{\bar{f}}$	the cluster containing all possible translations for the source phrase $\bar{f}$
c	a candidate translation in $\Omega_{\bar{f}}$
$C_{\bar{f}}$	the number of candidate translations in $\Omega_{\bar{f}}$
$\mu(\bar{f}_i\|\bar{e}_{j_i})$	the phrase translation probability distribution
$\phi(\mathbf{f})$	the feature vector for the source sentence $\mathbf{f}$
$\varphi(\mathbf{e})$	the feature vector for the target sentence $\mathbf{e}$
$\mathbf{W}$	a matrix-represented linear operator
$\|\mathbf{W}\|_{\mathcal{F}}^2$	Frobenius norm of the matrix-represented operator $\mathbf{W}$, which is defined by $\|\mathbf{W}\|_{\mathcal{F}}^2 = \mathbf{tr}(\mathbf{W}^T\mathbf{W})$
dim	the dimension of

TABLE 2.1: Notations used in this chapter.

2.1 Introduction

The field of *machine translation* (MT) is a sub–field of computational linguistics that attempts to automate all, or part of the process to translate text or speech from one natural language to another (Arnold et al., 1994).

The idea of machine translation can be traced back to the 1950*s*. In 1946, A. D. Booth and possibly others proposed using digital computers for translation of natural languages (Hutchins, 1997). In 1954, the Georgetown experiment (Slocum, 1985) involved fully–automatic translation of over 50 Russian sentences into English. Compared with human translations, the beginning of the research was very successful in terms of translation speed. Therefore, the original goal was the automatic translation of all kinds of documents at a quality equalling that of the best human translators (namely *fully automated high quality translation*) (Sager, 1988).

The real progress was much slower, however, and it became apparent very soon that this goal was unreachable in the foreseeable future. As an alternative, the researchers found that for many purposes the crude (unedited) machine translations could be useful to those who wanted to get a general idea of the content of a text in an unknown language as quickly as possible. The utilisation of this "side–effect" is proved to be more applicable than the original goal: between the 1960s and the 1980s, machine translation with human assistance (namely *human–aided machine translation*) has been a cost–effective option for multilingual bodies (e.g. the European Union) (Willée et al., 2002), that is, the MT systems produce rough translations which are then revised (post-edited) by translators. Since humans' post–editing to an acceptable quality could be expensive, many organisations are still putting effort on improving MT performance by traditional or cutting-edge technologies.

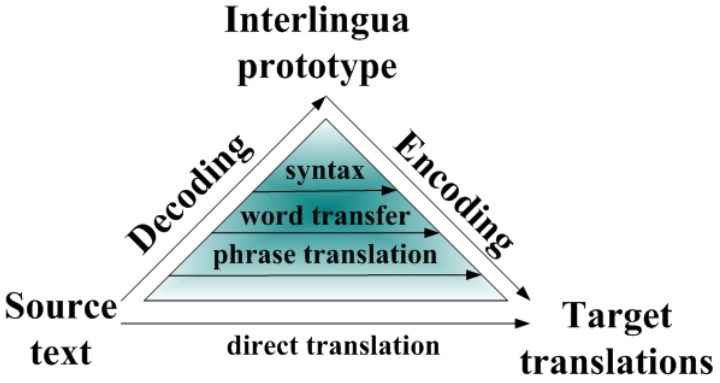


FIGURE 2.1: A diagram of the translation process.

In general, a bilingual translation process involves two stages (see Figure 2.1)

1. Decoding the meaning of the *source* text;

2. Re–encoding this meaning in the *target* language.

Behind this procedure lies a set of complex operations. To decode the meaning of the source text in its entirety, the translator must have full knowledge of the morphology, syntax and semantics, etc., of the source language, as well as the context features of the text. Meanwhile, the translator needs the same in–depth knowledge to re–encode the meaning in the target language.

All of these lead to two main challenges in machine translation: how to teach a computer to *learn* the context and syntax information from given texts and how to make it able to combine this information to *generate* an appropriate translation in the target language.

With different points of view in learning the context and syntax, a variety of methodologies have been proposed to MT. In order to appreciate the differences among these MT systems, they are classified into four categories, *rule–based machine translation* (RBMT), *example–based machine translation* (EBMT), *statistical machine translation* (SMT) and other machine translation systems.

The following sections outline these four categories in the order of their occurrence. Limited to our research interest, this thesis will concentrate on the SMT field, not only because of its popularity among the MT approaches, but also because of its close relationship to machine learning. Without loss of generality, the following description of the MT systems will centre around one task in machine translation: *sentence translation*, which involves translating a single sentence from one language to another.

2.2 Rule–based machine translation

A rule–based machine translation system is described as "meaning–oriented MT in an interlingua paradigm" (Hutchins, 1995). This category includes *transfer–based MT*, *interlingua MT* and *dictionary–based MT*, which have the same core principle: in order to create the correct translation it is necessary to have certain *intermediate representations* that capture extensive information of the original sentence. In general, there are three intermediate representations used in a rule–based MT system: the morphology, the syntax and the semantics patterns, which are collected and revised manually by expert translators. One example of translation process of an RBMT system is displayed in Figure 2.2 (Kaji, 1987). Given a source sentence, five steps are carried out to generate the target translation:

1. Translate words in isolation (*Morphological analysis*).
2. Words are then related to each other using syntax (*Syntactics analysis*).
3. Disambiguate words by the semantic features (*Semantics analysis*).

4. Transform the relationships between words and generate the syntactic trees (*semantics transformation* and *syntactic generation*).

5. Create the target translation (finalised by *Morphological synthesis*).

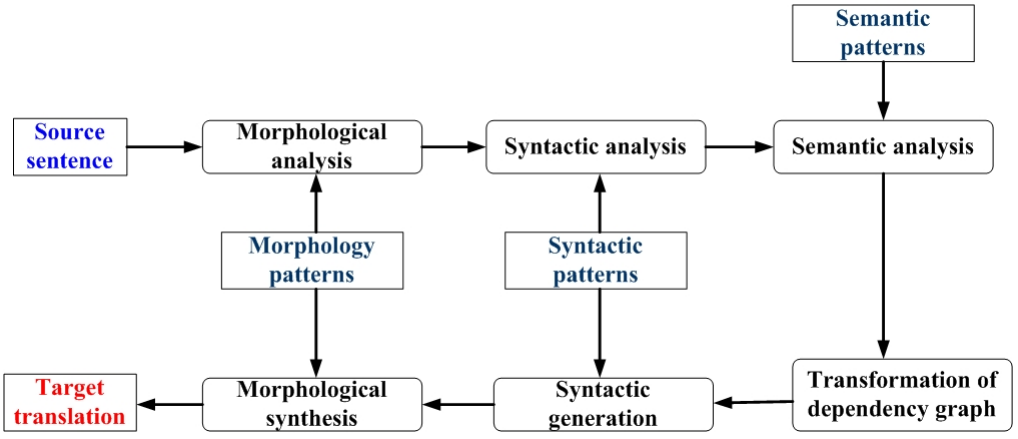


FIGURE 2.2: The translation process of an RBMT system. The morphological, the syntax and the semantics pattern databases are collected and revised manually by expert translators.

The rule–based approach is an effective way of implementing an MT system because of its extensibility and maintainability. Since much of the linguistic knowledge is static and assembling linguistic rules is a cumulative process, an RBMT system can be expanded and maintained as time goes on. Before the 1990s, RBMT systems were the dominant paradigm. During the 1980s, the methodology advanced rapidly on many fronts and several systems, such as SYSTRAN[1], LOGOS (Scott, 1994), EUROTRA (Arnold and des Tombe, 1987) and some Japanese systems (Nakamura et al., 1984; Kaji, 1987), achieved a great commercial success (Hutchins, 1995).

Nevertheless, an RBMT system is only a "semi-automatic" translation tool that requires both computational linguistics and skilled labour. It has two major limitations:

- *The construction of linguistic rules.* Since translating with an RBMT system involves applying a batch of linguistic rules, more precise rules usually result in better translations. However, translation rules are difficult to formulate in some special cases, making it hard for the system to deal with ambiguity. Take the case "idiom" for example, the phrase "all heart" in English should not be translated into "sou you de xin" (all of the hearts) in Chinese but "kang kai" (generous), which might conflict with the existing word-to-word translation rules and require introducing new idiomatic rules. However, collecting and revising these rules from

[1] `http://pages.unibas.ch/LIlab/staff/tenhacken/Applied-CL/3_Systran/3_Systran.html#history`

different corpora are laborious[2] compared with MT systems based purely on statistical methods, which will be discussed later in the chapter.

- *Processing efficiency.* In RBMT systems, a grammar consists of a lot of rewriting rules. Translation is then carried out by repeating pattern matching and transformation of tree or graph structures that represent the syntax and semantics of a sentence. As more and more knowledge is built into the system, managing the resulting complexity of the system becomes a vital issue (e.g. solving conflicts between existing rules and finding a pattern from the evergrowing rule database).

In order to address the latter problem, a number of methods have been developed, which involve the design of faster pattern matching algorithms (Forgy, 1982) and more effective rule database structures (Kaji, 1988). However, the most time-consuming aspect is the former: building linguistic rules. Although a machine can extract rough linguistic rules quickly from the corpora, these rules are far from satisfactory. Since the translation performance depends so heavily on precise rule representations, their qualities cannot be sacrificed. Therefore, a compromised solution, as discussed in (Gerber and Yang, 1997), is that rule extraction is automated as much as possible, while adhering to quality assurance by machine–aided manual validation. Unfortunately, this proposal was still expensive[3] compared with new and emerging MT methodologies and RBMT became unfashionable.

Although the dominance of the rule–based approach has been broken by the emergence of new MT methodologies, the linguistic rules remain useful for highly regular translations (e.g. numbers, dates). This fact results in the new utilisations of RBMT, such as the hybrid MT system (Sánchez-Martínez et al., 2009) and the RBMT system with automatic post-editing strategy (Dugast et al., 2007). At the same time, the existing RBMT systems continue to explore further avenues to automate high–quality linguistic architecture and still survive in the field of machine translation.

2.3 Example–based machine translation

The early 1990s was a major turning point for machine translation. New advanced techniques which are now loosely called "corpus–based" methods emerged and grew rapidly. The proposed corpus–based MT quickly replaced RBMT as the mainstream. In

[2] "The early process was laboriously manual: lexicographers checked entries to be added against a paper printout of the existing dictionary.....In those days, a lexicographer who could average 50 entries per day ($\sim 1,000$ per month) was a marvel!" (quoted from (Gerber and Yang, 1997))

[3] Unfortunately, we can not find any document providing a quantitative measurement of the construction time of the whole linguistic rule database. However, we quote a piece of the statement in (Gerber and Yang, 1997) to give the readers a guesstimate of the speed: "These days, it is not unheard of a lexicographer to generate $5,000$ entries per month". Obviously this speed is much lower than that of a statistical machine translation system.

corpus–based MT, one popular set of approaches is called *example–based machine translation* (EBMT), or alternative names by individual authors such as "analogy–based", "memory–based", "case–based" and "experience–guided" machine translation (Somers, 1999).

EBMT is an MT methodology that is characterised by the use of analogy in translations. The core of the theory is succinctly captured by Nagao, in his much quoted statement:

"Man does not translate a simple sentence by doing deep linguistic analysis, rather, man does translation, first, by properly decomposing an input sentence into certain fragmental phrases ..., then by translating these phrases into other language phrases, and finally by properly composing these fragmental translations into one long sentence. The translation of each fragmental phrase will be done by the analogy translation principle with proper examples as its reference." (Nagao, 1984)

In other words, this methodology attempts to mimic the cognitive process of human translators for the purpose of automating the translation process. As shown in Figure 2.3, there are three main tasks in EBMT:

- Match phrases of the source sentence against existing examples in the example database (Phrase Matching).
- Identify the corresponding translations for the matched examples (Phrase Alignment).
- Recombine these translations and generate the target translation (Phrase Recombination).

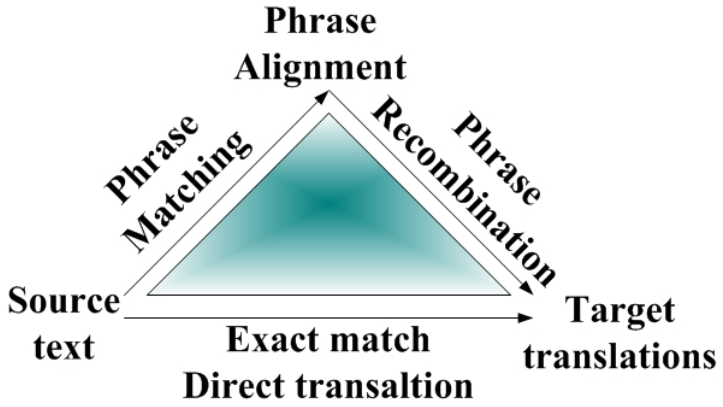


FIGURE 2.3: The translation process of an EBMT system (Figure is from (Somers, 1999)). The "translation pyramid" is similar to Figure 2.1 where the phrase translation, the word transfer and the syntactic analysis are all replaced by a single process – Phrase Alignment.

There is an affinity between EBMT and another corpus–based MT methodology – *statistical machine translation* (SMT), which will be elaborated in Section 2.4. Since this

thesis concentrates on the SMT field, this section just briefly describes these tasks and points out their relationships to SMT. The readers are referred to (Somers, 1999; Kit et al., 2003; Hutchins, 2005) and Section 2.4 for the techniques applied to EBMT.

2.3.1 Phrase matching

The *phrase matching* task is to segment the source sentence into a sequence of phrases (examples) that were stored in the example database. Usually, there can be more than one way to decompose a sentence in terms of a given set of known examples and thus the problem falls in finding a good criterion of the decomposition.

One method is to use the Viterbi–best sequence that provides the most probable path in generating the source sentence (Kit et al., 2003). However, there is no evidence showing that the most probable source path will generate the most probable translation in the target language.

The phrase matching task is similar to sentence decomposition in an SMT decoder (described in Section 2.4.6). But compared with an SMT decoder that utilises all possible sequences of phrases from the source sentence, the phrase matching task in EBMT is primitive.

2.3.2 Phrase alignment

The *phrase alignment* task is also known as "example acquisition", that attempts to extract reliable phrase pairs (example pairs) from the bilingual corpora. Although manual alignment by expert translators can produce quite reliable example pairs, there is a price to pay for the precision: the speed is far from enough in handling a corpus of hundreds of thousands of sentences. Therefore, automatic text alignment technology is essential.

As summarised in (Kit et al., 2003), there are two types of approaches for text alignment. One is the resource-poor approach, that relies mainly on sentence length statistics, co-occurrence statistics and some limited lexical information (Brown et al., 1991). A typical example is the word alignment approach proposed in (Och and Ney, 2003). The other is the resource-rich approach, which makes use of whatever is available and useful to facilitate the alignments, for example, the bilingual lexicon, the glossary and the syntax information (Wu, 2010).

This task is exactly the same as what a *phrase translation probability* model does in an SMT system (described Section 2.4.2). The reliability of example pairs extracted highly depends on the alignment information provided by the corpora as well as the text alignment technology used. So far, there is no alignment technology available to provide perfect example pairs, where heuristics are still employed to improve the performance (Och and Ney, 2003).

2.3.3 Phrase recombination

Using phrase matching, the source sentence is segmented into a sequence of source examples, whose translations are then acquired by phrase alignment. The final task, *phrase recombination*, is about how to make use of existing phrase translations to build up the target text. This is recognised as the most difficult step in EBMT but "has received considerably less attention" (Somers, 1999).

Since different languages have different syntax in modelling sentence structure and word order, translating phrases one-by-one usually results in an awful translation. Hence, phrase recombination should consider two key points:

- How to pick up the correct translation for a source phrase in a specific sentence.
- How to adjust the orders of the translations to form a smoothly readable sentence in the target language.

The former problem is the same as phrase prediction, which will be discussed in Section 2.4.2. For the latter problem, the syntactic structure can help re-organise the target translation (Turian et al., 2007), which however, involves some additional complicated natural language processing (NLP) tasks. Alternatively, a language model as used in an SMT system (described in Section 2.4.3) is also helpful. But translations generated by either strategy are not perfect, posing a challenge to MT researchers.

In effect, phrase recombination does the same task as an SMT decoder, while the steps are implemented in quite a different way.

Overall, EBMT and SMT are much alike and they share a lot of frameworks and learning techniques. It is even argued in (Somers, 1999) that SMT is also an example–based approach.

Compared with RBMT systems, the EBMT methodology is significantly faster. However, it does not guarantee better translation qualities: e.g. in (Gough and Way, 2004a) the performance of EBMT was inferior to that of RBMT, while in contrast Gough and Way (2004b) claimed that the former outperformed the latter. To the best of our knowledge, there are relatively few comparisons between EBMT and SMT and existing results (Groves and Way, 2005) have shown that EBMT is slightly inferior in translation qualities. Nevertheless, EBMT has been an active research interest as long as SMT, and today they still develop parallelly in the MT field.

Although they are now dominant paradigms, neither of them is able to generate perfect translations. A series of problems that prevent their achievements of high-quality translations will be pointed out in the rest of this chapter, where new methodologies are required so as to improve these MT systems to a new level.

2.4 Statistical machine translation

In 1988, a group from IBM published the preliminary results on a system based purely on statistical methods (Cocke et al., 1988). The effectiveness of this system was a considerable surprise at that time: almost half of the phrases were translated "correctly" (either matched exactly the translations in the corpus or expressed the same sense in slightly different words) and the processing speed was much higher than a rule–based approach. This experiment started a new era of machine translation and inspired other researchers to experiment with statistical methods of various kinds in subsequent years.

These statistics–based approaches to MT were then termed as *statistical machine translation* (SMT): the translations are generated on the basis of statistical models whose parameters are derived from the analysis of training corpora. Among the recent MT methodologies, SMT receives lots of positive reviews and becomes one of the dominant paradigms in machine translation.

2.4.1 The framework of SMT

The first ideas of *statistical machine translation* (SMT) were introduced by W. Weaver, who linked machine translation with statistical information theory (Weaver, 1949). He took the view that a source sentence (string $\mathbf{f}$) is simply a target translation (string $\mathbf{e}$) which was coded into the "source code". This perspective "brings into the foreground an aspect of the matter that probably is absolutely basic – namely, the statistical character of the problem". Following this idea, Brown et al. (1993) assigned a number $p(\mathbf{e}|\mathbf{f})$ to each sentence pair $(\mathbf{f}, \mathbf{e})$ which was interpreted as the probability that a translator, when presented with $\mathbf{f}$, would produce $\mathbf{e}$ as the translation. In this sense, SMT is thought of as a task where each source sentence $\mathbf{f}$ is translated into a target sentence $\mathbf{e}$, by means of a stochastic process that maximises the chance of obtaining the best translation

$$\mathbf{e} = \arg\max_{\hat{\mathbf{e}} \in \mathbb{E}} \left\{ p(\hat{\mathbf{e}}|\mathbf{f}) \right\} = \arg\max_{\hat{\mathbf{e}} \in \mathbb{E}} \left\{ \frac{p(\mathbf{f}|\hat{\mathbf{e}}) p(\hat{\mathbf{e}})}{p(\mathbf{f})} \right\} \tag{2.1}$$

in which $\mathbb{E}$ denotes the target translation pool. Since the denominator only depends on the source string $\mathbf{f}$, it is usually discarded and the solution is also represented as

$$\mathbf{e} = \arg\max_{\hat{\mathbf{e}} \in \mathbb{E}} \left\{ p(\hat{\mathbf{e}}|\mathbf{f}) \right\} = \arg\max_{\hat{\mathbf{e}} \in \mathbb{E}} \left\{ p(\mathbf{f}|\hat{\mathbf{e}}) p(\hat{\mathbf{e}}) \right\}. \tag{2.2}$$

The application of Bayes' rule to (2.1) is to divide the probability $p(\mathbf{e}|\mathbf{f})$ into two different sub-models: $p(\mathbf{f}|\hat{\mathbf{e}})$ represents the probability that the source string $\mathbf{f}$ is the translation of the target string $\hat{\mathbf{e}}$ and is referred to as the *translation model*; while the probability of observing that target string $p(\hat{\mathbf{e}})$ is referred to as the *language model*. These models mimic the cognitive process of human translators (see Figure 2.1): a translator proceeds

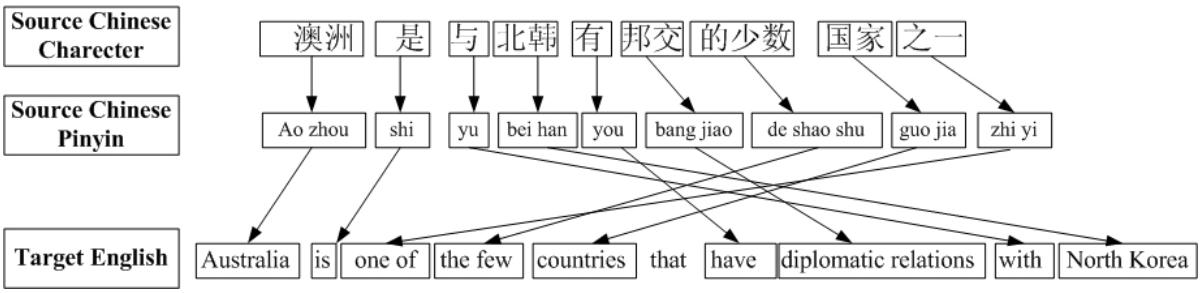


FIGURE 2.4: The process of phrase–based translation. The source (Chinese) sentence is segmented into phrases, each of which is translated into a target (English) phrase. Note that the target phrases may be reordered.

by first understanding the source sentence, and then expressing the context in the target language that he has thus grasped. In effect, the division of $p(\mathbf{e}|\mathbf{f})$ allows the balance of the translation and the fluency qualities, which will be discussed in Section 2.4.2 and Section 2.4.3 respectively.

However, these two components are far from enough in modelling sentence translation in practice. Take Figure 2.4 for example, the translation of a sentence is not merely a combination of word-to-word translations, it has word (or phrase) reorderings. This phenomenon adds load to the language model $p(\mathbf{e})$ and makes it difficult to approximate the entire re-encoding process in Figure 2.1. To achieve an acceptable quality of translation, a further division of these models into more specific ones is essential.

To enable a richer set of sub-models, the posterior probability $p(\mathbf{e}|\mathbf{f})$ is generally modelled with a log–linear maximum entropy framework (Berger et al., 1996), which makes it easier to introduce additional models. Under this framework, the target translation is given by the formula

$$\mathbf{e} = \arg\max_{\hat{\mathbf{e}} \in \mathbb{E}} \left\{ \exp\left(\sum_{m} \lambda_m h_m(\hat{\mathbf{e}}, \mathbf{f}) \right) \right\} \tag{2.3}$$

which uses a number of feature functions h_m to represent individual models and λ_m are the corresponding scale factors.

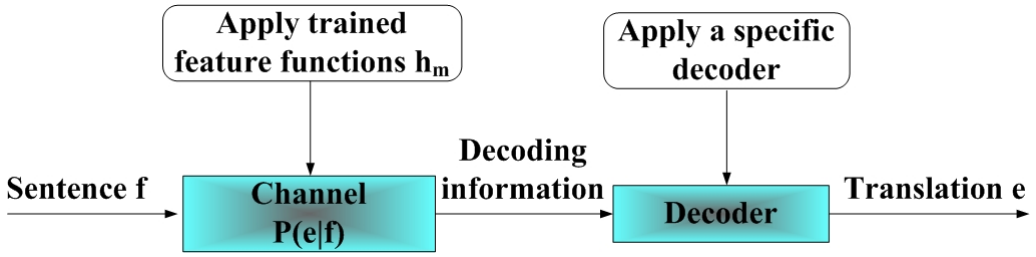


FIGURE 2.5: The translation procedure.

The target translation $\mathbf{e}$ is then generated by a decoder (described in Section 2.4.6) to maximise the score function in (2.3). Figure 2.5 depicts a simplified procedure of

SMT, where probability $p(\mathbf{e}|\mathbf{f})$ is now characterised by a set of sub-models, which work together to provide the decoding information.

To make concepts clear, we outline the score function used in the state-of-the-art SMT system – MOSES (Koehn et al., 2005)

$$\mathbf{e} = \arg\max_{\hat{\mathbf{e}}\in\mathbb{E}}\{p_t(\mathbf{f}|\hat{\mathbf{e}})^{\lambda_T} p_{lm}(\hat{\mathbf{e}})^{\lambda_{lm}} p_{lex}(\mathbf{f}|\hat{\mathbf{e}})^{\lambda_{lex}} p_d(\mathbf{f},\hat{\mathbf{e}})^{\lambda_d} w^{|\hat{\mathbf{e}}|\lambda_w}\} \tag{2.4}$$

The system is made up of five sub-models: the *translation model* $p_t(\mathbf{f}|\hat{\mathbf{e}})$ and the *language model* $p_{lm}(\hat{\mathbf{e}})$ mentioned above, as well as a *lexical weight translation* model used as a supplement of $p_t(\mathbf{f}|\hat{\mathbf{e}})$; a *phrase reordering model* $p_d(\mathbf{f},\hat{\mathbf{e}})$ that is used to improve the fluency of translation when combining target phrases; and a word penalty $\omega^{|\hat{\mathbf{e}}|}$ ($|\hat{\mathbf{e}}|$ returns the number of words in the target translation $\hat{\mathbf{e}}$) that controls the length of the output translation. Each part is weighted by a scale factor λ_m that can be pre-defined (Koehn, 2004) or trained with respect to the final translation quality measured by an error criterion (Och, 2003). The framework of this SMT system is illustrated in Figure 2.6, where the original decomposition has been greatly modified, with more elaborate models and greater capacity in modelling the translation process.

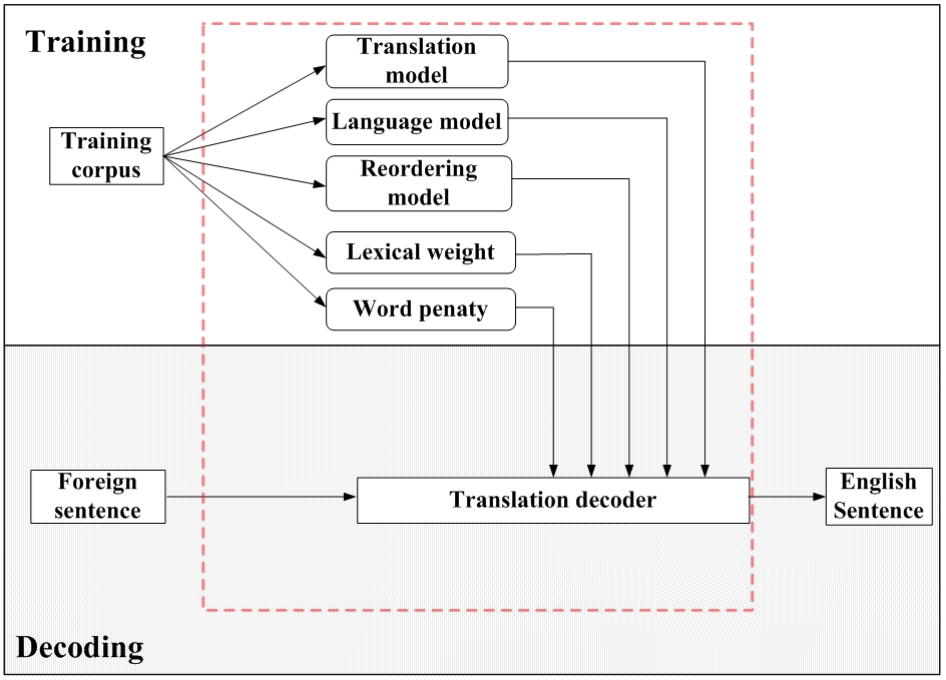


FIGURE 2.6: The training (top box) and the decoding (shaded box) procedures for an SMT system. There are five sub-models that function together for generating the combined decoding information.

Since SMT is the baseline approach in our MT experiments throughout the thesis, the details of these sub-models as well as their developments are discussed in the rest of this section.

2.4.2 Phrase translation probability model

During the past few years, MT systems tended to use sequences of words, namely *phrases*, as the decoding prototypes so as to capture the word context in the language corpora. Figure 2.4 illustrates an example of the phrase–based translation. The source sentence (Chinese) is segmented into a set of consecutive words (phrases), which are translated in the target language. The target phrases then may either stay in the same position or be reordered to produce a fluent translation.

Mathematically, a source sentence $\mathbf{f}$ is segmented into a sequence of I phrases $\bar{\mathbf{f}}^I$. Each source phrase $\bar{f}_i \in \bar{\mathbf{f}}^I$ is translated into a target phrase $\bar{e}_{j_i} \in \bar{\mathbf{e}}^I$, where its position may be reordered (i.e. $j_i \neq i$) afterwards due to the grammar. Following the assumption that the target translations are conditionally independent given their source phrases (Koehn, 2003), the phrase translation is generally modelled by a *phrase translation probability* distribution $\mu(\bar{f}_i|\bar{e}_{j_i})$. In this case the probability of $\mathbf{f}$ given $\mathbf{e}$ is decomposed into a product of phrase translation probabilities

$$p_t(\mathbf{f}|\mathbf{e}) := p_t(\bar{\mathbf{f}}^I|\bar{\mathbf{e}}^I) = \prod_{i=1}^{I} \mu(\bar{f}_i|\bar{e}_{j_i}) \tag{2.5}$$

This phrase translation probability (PTP) model has changed little among different SMT systems. It contains two stages: *phrase pair extraction* and *modelling the phrase translation probabilities*.

2.4.2.1 Phrase pair extraction

The procedure starts by extracting a phrase translation table from a parallel corpora based on bilingual word alignment. At this point, the tool commonly used to establish word alignments is *GIZA++* toolkit (Och and Ney, 2003), which is an implementation of a set of statistical models (namely IBM Models) for generating a Viterbi–best word alignment sequence.

Since bilingual word alignment is usually carried out in both directions (i.e. source–to–target and target–to–source), there are three existing approaches to extract phrase pairs:

- all alignment points of the intersection of the two alignments;
- the points of the union of the two alignments;
- start with the intersection and add additional alignment points based on a heuristic procedure (Och and Ney, 2003; Koehn, 2003).

As shown in (Och and Ney, 2003), the phrase pairs extracted from the intersection of the two alignments are inadequate while those extracted from the union contain too many inaccurate ones. Alternatively, the heuristic approach seems to be a good compromise and has been used in most of the SMT systems.

2.4.2.2 Modelling the phrase translation probabilities

Suppose for a source phrase $\bar{f}$, there exists a set of target translation candidates $c \in \Omega_{\bar{f}}$ in the phrase table and the number of candidates is $C_{\bar{f}}$. Let us denote the frequency of phrase pair $(\bar{f}, c)$ as $count(\bar{f}, c)$ and assign $(\bar{f}, c)$ a discrete distribution $\mu(\bar{f}|c)$. Then the likelihood to see phrase pair $(\bar{f}, c)$ with $count(\bar{f}, c)$ times is

$$\mathcal{L}\big(count(\bar{f}, c) \mid \mu(\bar{f}|c)\big) = \left(\begin{array}{c} count(\bar{f}, c) \\ \sum\limits_{c'} count(\bar{f}, c') \end{array} \right) \mu(\bar{f}|c)^{count(\bar{f},c)} (1 - \mu(\bar{f}|c))^{\sum\limits_{c' \neq c} count(\bar{f},c')} \tag{2.6}$$

Taking the derivative of $\mathcal{L}$ with respective to $\mu(\bar{f}|c)$ and setting it equal to zero, we arrive at the *maximum likelihood estimation* (MLE) of the phrase translation probability

$$\mu(\bar{f}|c) = \frac{count(\bar{f}, c)}{\sum\limits_{c'} count(\bar{f}, c')} \tag{2.7}$$

This pure statistical method is commonly used in current state–of–the–art SMT systems (Och and Ney, 2003; Koehn, 2004; Koehn et al., 2005; Zens and Ney, 2006), although it has three major limitations

- The phrase translation probability only depends on the frequency of the phrase pairs; the sentence context (e.g. lemma and topical information) in which phrases occur is completely ignored.
- For the low–frequent phrase pairs, the variance of MLE could be considerably large (Bishop, 2006), making the prediction over–fit the training data.
- The use of this probability makes the SMT decoder (describe in Section 2.4.6) biased towards longer phrases.

To tackle these problems, current systems usually combine some heuristic functions, such as bi–directional phrase translation probabilities and lexical weights (Koehn et al., 2005). But to better characterise the phrase translation probability (PTP) distribution, more sophisticated methods are required.

As machine learning techniques become more and more attractive in the MT field, several discriminative methods have been proposed and applied for the PTP model. To the best

of our knowledge, Vickrey et al. (2005) started the research by applying a *word sense disambiguation* (WSD) model, that learnt word translation based on the context, syntax information and lemma. Basically, they set up a logistic regression model to encode the conditional probability $\mu(\bar{f}|c)$

$$\mu(\bar{f}|c) = \frac{1}{Z}\exp\{\mathbf{w}^T\phi(\bar{f},c)\}$$

with partition function $Z = \sum_{c'\in\Omega_{\bar{f}}} \exp\{\mathbf{w}^T\phi(\bar{f},c')\}$, weight vector $\mathbf{w}$ and feature vector $\phi(\bar{f},c)$. The goal is to maximise the same objective function (2.5), which they maximised by maximising its logarithm (the log–conditional–likelihood).

A big leap of encoding the probability distribution $\mu(\bar{f}|c)$ as a logistic regression function is the cooperation with the feature vector $\phi(\bar{f},c)$, making the solution more flexible than just counting the frequency of the phrase pairs as used in equation (2.7). Since the feature vector can be constructed from the sentence context and syntax information, such as the occurrence of word sequences and part–of–speech (POS) tags, the resulting conditional probabilities are more capable in word disambiguation.

However, the work in (Vickrey et al., 2005) just contained a crude application to machine translation: $\bar{f}$ and c were merely words rather than phrases, and the experiments were carried out only for word blank–filling tasks, not sentence translation.

By extending from words to phrases, Carpuat and Wu (2007) used an ensemble of four combined WSD models – a *naive bayes* model, a *maximum entropy* model, a *boosting* model and a *kernel PCA-based* model – to predict phrase translation probabilities. The average of the predictions was then passed to the SMT decoder to help phrase disambiguation. The resulting MT system outperformed a traditional SMT system (Koehn, 2004), confirming the advantages brought by discriminative training. Nevertheless, Carpuat and Wu (2007) only used $40,000$ sentence pairs as the training set, which is rather small compared to the standard in the discipline (e.g. the EuroParl corpus commonly used for training contains 1 million sentences). Whether the model scales up to larger data collections was not discussed.

Alternatively, Giménez and Màrquez (2007) proposed a *context–aware discriminative model* for sentence translation. Similar to (Carpuat and Wu, 2007), Giménez and Màrquez (2007) jumped out of the probability framework and dealt with phrase translation using a classification scheme. Instead of considering MLE for each $(\bar{f},c)$, they regarded the translation for a unique source phrase $\bar{f}$ as a multi–class classification problem and introduced a set of *support vector machines* (SVMs) to perform phrase disambiguation. Since an SVM is a binary classifier, the simple *one–versus–all* strategy was applied and the SVM confidences were then transformed into the phrase translation

probabilities with the soft–max function (Bishop, 2006)

$$\mu(\bar{f}|c) := \frac{1}{Z}\exp\{SCORE_{SVM}(\bar{f}, c)\} \tag{2.8}$$

where $Z = \sum_{c' \in \Omega_{\bar{f}}} \exp\{SCORE_{SVM}(\bar{f}, c')\}$ is the partition function.

This delicate discriminative model considers a wider feature context and imposes regularisations in a function space to guard against over-fitting. It provides a useful mechanism in order to improve phrase disambiguation for current phrase–based SMT systems. However, the work itself is not complete for machine translation. As Giménez and Màrquez (2007) concluded at the end of their work, there are several open questions that were not solved:

1. Because of the time complexity of SVMs, the phrase table used is limited to a reduced set of "frequent" phrases.
2. The reordering and word penalty models in an SMT system are disabled due to the integration problem of the SVMs and other models.
3. Although they have considered the sentence context, the potential connections between target translations for the same source phrase are still ignored.

As a conclusion, the PTP model in an SMT system still has great potential for development. For instance, phrase translation can be regarded as a problem beyond multi-class classification. Figure 2.7 depicts an English-to-Chinese example, where the Chinese translations of the English word "wear" are connected via certain latent structure. In this case, better disambiguation performance is achievable if the PTP model respects this structure, where the newly emerging structured learning technologies in ML may find a critical role to play. In Chapter 5, we will show our work for this model, that combines new ML technologies with a traditional MT mechanism.

2.4.3 Language model

The *language model* $p_{lm}(\mathbf{e})$ provides the probabilities of target word sequences, which aim at improving the fluency of target translations. Since estimating sequence probability is difficult in sentences in which phrases can be arbitrarily long and hence some sequences are not observed during model training, the model is usually approximated by an n–gram conditional probability model (see Figure 2.8). The computation of this

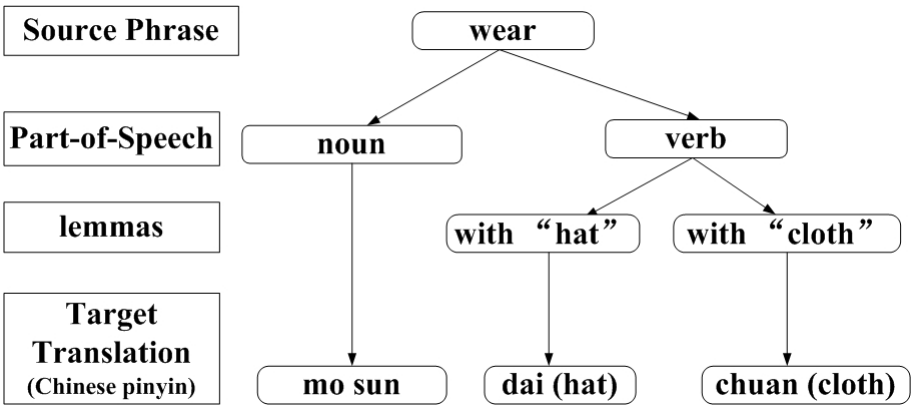


FIGURE 2.7: An English-to-Chinese word translation example, where the Chinese translations are connected via a latent structure characterised by part-of-speech and lemma.

P(*look, before, you, leap*)≈ p(*look*)p(*before | look*)p(*you | look, before*)p(*leap | before, you*)

FIGURE 2.8: A tri-gram language model for the sentence "Look before you leap".

sequence probability is given by formula

$$\begin{aligned} p_{lm}(\mathbf{e}) &= \prod_{i=1}^{|\mathbf{e}|} p_{lm}(e_i|e_{i-1},\ldots,e_1) \\ &\approx p_{lm}(e_1)p_{lm}(e_2|e_1)\cdots p_{lm}(e_{n-1}|e_{n-2},\ldots,e_1)\prod_{i=n}^{|\mathbf{e}|} p_{lm}(e_i|e_{i-1},\ldots,e_{i-n+1}) \end{aligned} \tag{2.9}$$

with the assumption that the probability of observing the i-th word e_i in the context history of the preceding $i-1$ words can be approximated by the probability of observing it in the shortened context history of the preceding $n-1$ words.

Theoretically this conditional probability $p_{lm}(e_i|e_{i-1},\ldots,e_{i-n+1})$ is easy to achieve by maximum likelihood estimation

$$p_{lm}(e_i|e_{i-1},\ldots,e_{i-n+1}) = \frac{count(e_i,e_{i-1},\ldots,e_{i-n+1})}{count(e_{i-1},\ldots,e_{i-n+1})}$$

But in practice, MLE biases high for observed n–grams and biases low for unobserved ones. In particular, if a given n–gram $(e_i,e_{i-1},\ldots,e_{i-n+1})$ has not been observed in the training data, MLE returns 0 probability that may destroy the decoding of a new sentence (the sequence probability is always 0). To tackle these problems, different heuristics have been applied to make the model more friendly to the data.

Developing the language model has become a fruitful research topic, which is used in many NLP applications such as speech recognition, part–of–speech tagging and machine translation. The readers are referred to (Ponte and Croft, 1998) for the construction of an n–gram language model. Two implementations: the CMU–Cambridge language

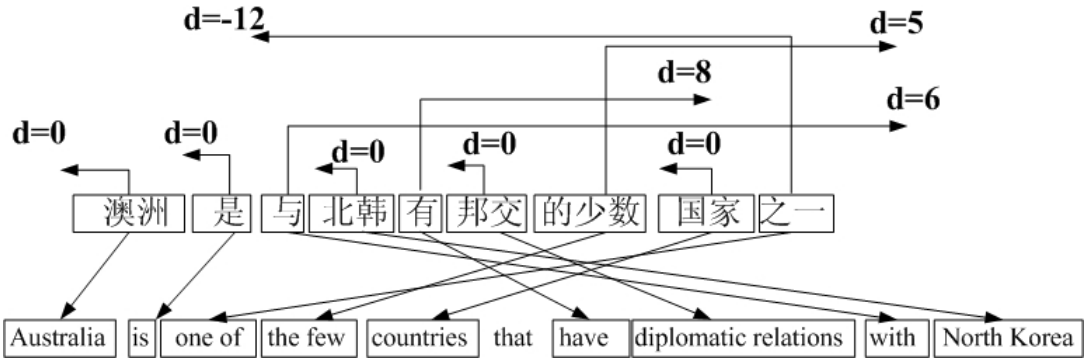


FIGURE 2.9: The word distance–based reordering model. Each phrase in the source sentence has a reordering distance d_i, and the total cost of phrase movements is computed by $p_d(\mathbf{f}, \mathbf{e}) = \alpha^{-\sum_i d_i}$.

model toolkit (Clarkson and Rosenfeld, 1997) and SRILM (Stolcke, 2002) are used in the proposed work throughout the thesis.

2.4.4 Phrase reordering model

Word or phrase reordering is a common problem in bilingual translations arising from different grammatical structures. For example, in French the adjective is often behind the corresponding noun, while when translated into English, "adjective before noun" is often the correct grammar. In general, learning the phrase movements in a source sentence to obtain correct target word orders is of great importance for improving the fluency of machine translation.

Taking a Chinese–to–English translation (see Figure 2.9) for example, obviously not all words are translated one by one and some words are translated far behind after its preceding words are translated (e.g. phrase "North Korea"). Therefore, an ideal phrase reordering model should be able to handle arbitrary distance phrase movements. That is, for I phrases building up the source sentence each of which has at most $C_{\bar{f}}$ translation candidates, the model should then consider $C_{\bar{f}}I!$ combinatorial possibilities. Unfortunately this ideal model is *NP–complete*, as discussed in (Knight, 1999).

During the development of SMT, the phrase reordering model has received more attention than other sub-models such as the PTP and the language models. A variety of algorithmic and heuristic techniques has been proposed and claimed to achieve better performance. According to the principles, we classify these approaches into three categories.

2.4.4.1 Generative phrase reordering models

The generative phrase reordering models generate phrase reordering probabilities using certain pre-defined criteria. As the phrase reordering problem is computationally expensive, a simple generative model, namely *word distance-based reordering* (WDR) model, is commonly used among some SMT systems (Och et al., 1999; Och, 2003; Koehn, 2004; Zens et al., 2005). This model defines the reordering distance for the i–th phrase $\bar{f}_i$ as (see Figure 2.9)

$$\begin{aligned} d_i \quad := \quad & \text{abs}(\text{last word position of previously translated source phrase} + 1 \\ & -\text{first word position of newly translated source phrase } \bar{f}_i) \end{aligned} \tag{2.10}$$

then the total cost of phrase movements $p_d(\mathbf{f}, \mathbf{e})$ in a sentence pair $(\mathbf{f}, \mathbf{e})$ is log-linearly proportional to these reordering distances

$$p_d(\mathbf{f}, \mathbf{e}) = \alpha^{-\sum_i d_i} \tag{2.11}$$

where α is a pre–defined constant.

Similarly, a flat reordering model (Wu, 1996; Zens et al., 2004) is also popular. The model assigns constant probabilities for the "monotone" ($d = 0$) and the "switching" ($d \neq 0$) orders

$$p_d(o) = \begin{cases} p & o = monotone \\ 1 - p & o = switching \end{cases}$$

where the probability p is pre-defined to favour monotone or non-monotone orientations based on the prior knowledge of the bilingual pairs. However, (Xiong et al., 2006) has shown that this model is inferior to the word distance-based reordering (WDR) model. We postulate this is because of the flat probability assigned to different distance reorderings, that is, the long–distance reorderings are as possible as the short ones. The lack of sensitivity in reordering distance finally causes wrong distance phrase movements and spoils the translations.

Although the above models are simple and time efficient, the content independence makes it difficult to capture many phrase movements caused by grammars. To tackle this problem, Tillmann (2004) and Koehn et al. (2005) developed a model that attempted to learn phrase reordering based on the contents – the *lexicalized reordering* (LR) model. The model splits the distance space into several segments (e.g. two segments, namely "monotone" and "switching" orders as described above), each of which represents a phrase reordering orientation o. Then it learns the local orientation probabilities for each bilingual phrase pair $b_i = (\bar{f}_i, \bar{e}_{j_i})$ using maximum likelihood estimation

$$p_d(o|b_i) = \frac{count(o, b_i)}{\sum_{o'} count(o', b_i)} \tag{2.12}$$

Finally the sub-model $p_d(\mathbf{f}, \mathbf{e}) = \sum_{b_i} p_d(o|b_i)$ represents the cumulative cost of phrase movements and is integrated into (2.3) to help the decoder correct the word orders in the target language. Improved translation results using this content dependent model have been reported in (Koehn et al., 2005). However, this MLE solution only characterises phrase movements by their frequencies, while in practice the movement of a phrase also depends on the sentence context. For example, given two nouns "A" and "B", the French phrase "A de B" is usually translated into "B A" in English. Hence, whether "B" is reordered or not can be better chariterised by capturing the word context "de" near "B".

2.4.4.2 Discriminative phrase reordering model

Adopting the idea of predicting the phrase reordering orientations, researchers started exploiting the context or grammatical contents which may relate to phrase reordering (Tillmann and Zhang, 2005; Xiong et al., 2006; Zens and Ney, 2006). In general, the distribution of phrase reordering is of the form

$$p_d(o|b_i, \mathbf{w}_o) = \frac{h(\mathbf{w}_o^T \phi(b_i, o))}{Z(b_i)} \tag{2.13}$$

where $\phi(b_i, o)$ is a feature vector based on the phrase pair b_i and the sentence context, $\mathbf{w}_o = [w_{o,0}, \ldots, w_{o,dim(\phi)}]^T$ is the weight vector measuring features' contribution to an orientation $o \in \mathcal{O}$, h is a pre-defined monotonic function (usually an exponential function) and the partition function Z is given to normalise the probabilities over the set of phrase reordering orientations $\mathcal{O}$

$$Z(b_i) = \sum_{o' \in \mathcal{O}} h(\mathbf{w}_{o'}^T \phi(b_i, o'))$$

A number of discriminative models have been proposed, benefiting from different ML methodologies. They differ in four aspects:

- The definition of the orientation set $\mathcal{O}$.
- The model used to learn $\{\mathbf{w}\}_{o \in \mathcal{O}}$.
- The phrase feature expression.
- The utilisation of these orientation probabilities.

For example, Zens and Ney (2006) proposed the *discriminative reordering model* in which the orientation set was a binary set $\mathcal{O} = \{d < 0, d \geq 0\}$ or a multi-class set

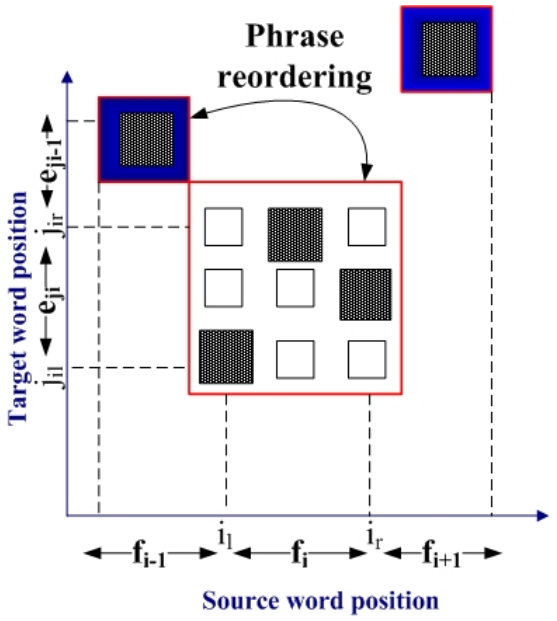


FIGURE 2.10: A reordering example of the phrase pair $(\bar{f}_i, \bar{e}_{j_i})$ (the word alignments are in black rectangle) and the linguistic environment around the source phrase (blue), where the word–level and the phrase–level features (e.g. phrases $\bar{f}_{i-1}$ and $\bar{f}_{i+1}$) are extracted.

$\mathcal{O} = \{d < -1, d = -1, d = 0, d = 1, d > 1\}$. The model was based on a *maximum entropy* (ME) framework

$$\max_{\{\mathbf{w}_o\}_{o\in\mathcal{O}}} \{-\sum_{b_i}\sum_{o\in\mathcal{O}} p_d(o|b_i, \mathbf{w}_o) \log p_d(o|b_i, \mathbf{w}_o)\} \tag{2.14}$$

to tune the feature parameters $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$ and applied the Generalised Iterative Scaling (GIS) algorithm (Darroch and Ratcliff, 1972) to derive the solution (see Table 2.2). During the decoding (described in Section 2.4.6), the reordering probabilities (2.13) are used directly in equation (2.4) to help the decoder find a Viterbi-best orientation sequence.

The advantage the model brings is the utilisation of word–level and phrase–level context features (see Figure 2.10) and syntax information (e.g. part-of-speech (POS)), with the purpose of better characterising phrase movements. Although this model was reported to produce improved classification performance over the lexicalized reordering (LR) model, there are two weaknesses restricting its development

- The simplified orientation set limits its ability to capture more complicated phrase reorderings. Theoretically the model can impose an arbitrary large orientation set, however the training times for ME models are relatively high when the output classes increase, making such development impractical.

Goal: minimise the entropy function (2.14).

Input: The samples $\{(o_i, \phi(b_i, o_i))\}$

1) **Initialise** a starting value $\mathbf{w}_o$ for $\{\mathbf{w}_o\}_{o \in \mathcal{O}}$

2) **for** each feature $\phi_j(\bullet, o)$ **do**

$N_{o,j}$ = frequency of $\phi_j(\bullet, o)$

3) **for** $t = 1 \cdots \tau$ **do**

for each feature $\phi_j(\bullet, o)$ **do**

$EXPECT\_VALUE\big(\phi_j(\bullet, o)\big) = 0$;

for each training data (phrase pair block) b_i **do**

for each class c **do**

$s[c] = \mathbf{w}_{o_c}^T \phi(b_i, o_c)$;

$z = \sum_c \exp\{s[c]\}$;

for each class c **do**

for each feature $\phi_j(\bullet, o_c)$ **do**

$EXPECT\_VALUE\big(\phi_j(\bullet, o_c)\big) += \phi_j(b_i, o_c) \exp\{s[c]\}/z$;

for each feature $\phi_j(\bullet, o)$ **do**

update $w_{o,j} += \frac{1}{C} \log \frac{N_{o,j}}{EXPECT\_VALUE\big(\phi_j(\bullet, o)\big)}$

until converge (small change in $\sum_{b_i} \log p_d(b_i, o_i)$)

Output: $\{\mathbf{w}_o\}_{o \in \mathcal{O}}$

TABLE 2.2: Pseudo-code of the Generalised Iterative Scaling (GIS) algorithm.

- All phrase features used are indicator functions. However, different features might have a different impact to various orientations, which can not be captured by the indicator functions.

Another approach employing the maximum entropy framework and indicator features is the *maximum entropy model* (MaxEnt) (Xiong et al., 2006). Slightly different from (2.13), the distribution of phrase reordering is expressed as

$$p_d(o|b_i, b_{i'}, \mathbf{w}_o) = \frac{h(\mathbf{w}_o^T \phi(b_i, b_{i'}, o))}{Z(b_i, b_{i'})}$$

which is conditioned by two adjacent phrase blocks b_i and $b_{i'}$. In this case, the model only requires two orientations $\mathcal{O} = \{b_i \text{ is on the left of } b_{i'},\ b_i \text{ is on the right of } b_{i'}\}$. But as a tradeoff, a CKY style decoder (described in Section 2.4.6) is required to retrieve the target translations. The resulting MT system produced similar improvements on two Chinese–to–English translation tasks, while the problems pointed out above remained unsolved. Furthermore, when b_i and $b_{i'}$ are long phrase pairs, there are few or no examples for $(b_i, b_{i'}, o)$ in the training set, causing a problem of data sparseness in this model.

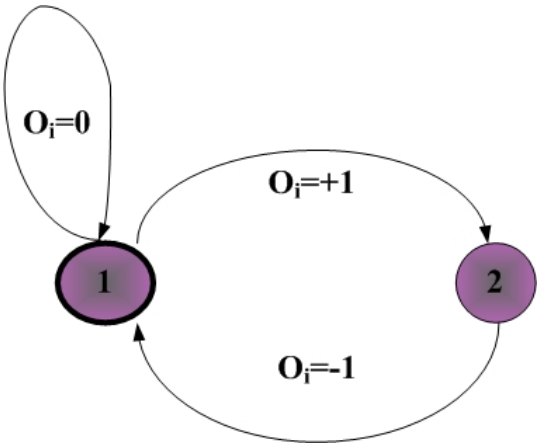


FIGURE 2.11: A first order Finite State Machine for phrase reordering.

The final method is the *localized prediction* model proposed in (Tillmann and Zhang, 2005), where the orientation probability is of the form

$$p_d(b_i, o_i | b_{i'}, o_{i'}, \mathbf{w}) = \frac{\exp(\mathbf{w}^T \phi(b_i, o_i; b_{i'}, o_{i'}))}{Z(b_i, o_i, b_{i'}, o_{i'})} \tag{2.15}$$

A new phrase pair block b_i in this model has only three orientations with respect to its predecessor $b_{i'}$:

$$\mathcal{O} = \{b_i \text{ is on the left of } b_{i'},\ b_i \text{ is on the right of } b_{i'},\ b_i \text{ and } b_{i'} \text{ are not adjacent}\}.$$

The first two classes are only active when b_i and $b_{i'}$ are adjacent, in other words, the target phrases $\bar{e}_{j_i}$ and $\bar{e}_{j_{i'}}$ as well as source phrase $\bar{f}_i$ and $\bar{f}_{i'}$ are adjacent.

The model brings a novel idea for feature expression. Apart from indicator functions, the model also considered real value features such as the lexicalized reordering probabilities $p_d(o|b_i)$, the tri-gram language model probabilities, the lexical weights (described in Section 2.4.5) and so forth. In this way, the probability of each localized phrase block b_i was computed by (2.15). A beam search decoder (described in Section 2.4.6) was then applied to pick up the blocks matching the current source phrase and build up a translation to maximise the product of block probabilities $p_d(b_1^I, o_1^I) = \prod_{i=1}^{I} p_d(b_i, o_i | b_{i-1}, o_{i-1})$.

The weakness of this phrase reordering model is that it can only consider the localized phrase reorderings, which is suitable for bilingual translation tasks where the word orders in one language are similar to those in another (e.g. Arabic–to–English). However, for the translation tasks which contain a lot of long distance phrase movements (e.g. Chinese–to–English), we doubt its effect on improving the fluency of translations.

2.4.4.3 Weighted finite state transducer

A phrase reordering model in the framework of weighted *finite state machine* (FSM) (e.g. Figure 2.11) is proposed in (Kumar and Byrne, 2005), in which the phrase reordering orientation o_i of a phrase pair $(\bar{f}_i, \bar{e}_{j_i})$ depends on the phrase pair itself and the preceding state ϕ_{i-1}:

$$p_d(o_i|\bar{f}_i, \bar{e}_{j_i}, \phi_{i-1}) = \begin{cases} w_1(\bar{f}_i, \bar{e}_{j_i}) & o_i = +1,\ \phi_{i-1} = 1 \\ 1 - w_1(\bar{f}_i, \bar{e}_{j_i}) & o_i = 0,\ \phi_{i-1} = 1 \\ 1 & o_i = -1,\ \phi_{i-1} = 2 \end{cases}$$

in which the orientation set is defined by $\mathcal{O} = \{-1 : d = -1;\ 0 : d = 0;\ 1 : d = 1\}$.

The above formula shows a first order FSM in modelling phrase movements and in their work a second order FSM was also constructed to capture more complicated phrase reorderings. The transit parameters $\mathbf{w}$ were then estimated using an EM–style method. However, the windows for phrase reordering are very restricted due to the computational expense, which limits its further development.

For modelling phrase movements, there is no doubt about the benefits from machine learning: compared with the traditional statistical methods used (e.g. the lexicalized reordering model), the utilisation of linguistic features and more robust learning agents ML brings usually results in better translation performance. However, the existing ML methods applied can not capture phrase movements perfectly, especially long distance reorderings. Furthermore, the researchers only reported improvements on the quality of translation, while few comments and case studies are given to the analysis of performance of phrase movements. Hence, the investigation of ML methods for phrase reordering as well as a detailed analysis of performance of phrase movements are still worthwhile. This belief motivates our development of new discriminative phrase reordering models, which will be described in Chapter 6.

2.4.5 Supplementary models: lexical weight translation, word penalty and phrase penalty

One major weakness of using phrases as prototypes is that the bias in target translations is usually upwards for long phrase pairs and downwards for shorter ones. This effect is easy to explain since the phrase translation probability $\mu(\bar{f}_i|\bar{e}_{j_i})$ always satisfies $0 < \mu(\bar{f}_i|\bar{e}_{j_i}) \leq 1$, therefore the fewer number of phrases in equation (2.5), the larger the probability $p_t(\mathbf{f}|\mathbf{e})$ will be.

To balance the usage of phrases with different length, a supplementary model, namely *lexical weight translation* model that makes use of word alignments, is applied. Let us denote $\bar{f}_i = [f_{i_l}, \ldots, f_{i_r}]$, $\bar{e}_{j_i} = [e_{j_{i_l}}, \ldots, e_{j_{i_r}}]$ and the word–level alignments for the

sentence pair $(\mathbf{f}, \mathbf{e})$ is $\mathbf{a} = \{(k,l)|f_k \text{ aligns with } e_l\}$, then the lexical weight of $(\bar{f}_i|\bar{e}_{j_i})$ is given by the formula (Koehn, 2003)

$$p_{lex}(\bar{f}_i|\bar{e}_{j_i}, \mathbf{a}) := \prod_{k=i_l}^{i_r} \frac{1}{|(k,l) \in \mathbf{a}|} \sum_{\forall(k,l)\in\mathbf{a}} p_t(f_k|e_l) \tag{2.16}$$

This re–scoring model aims at checking how well the words in a phrase pair translate to each other. Intuitively, if the words in $\bar{f}_i$ are not generally translated into the words in $\bar{e}_{j_i}$, then this phrase pair may just occur by chance and can not be a good candidate.

One similar problem occurs in the language model: an SMT system with a language model usually biases towards short translations. To tackle this problem, the *word penalty* $\omega^{|\mathbf{e}|}$ or the *phrase penalty* ω^I come out to be a simple mean to calibrate the output length. If $\omega > 1$ the system biases towards longer translations, otherwise it favours shorter ones. The factor ω (called *word cost* or *phrase cost*) is usually pre-defined or is optimised by cross-validating on a validation set.

2.4.6 Translation decoder

Given a source sentence, all its sub-phrases are extracted and their target translations are predicted using the phrase translation probability model. Then generating the target sentence is like piling up the building blocks (phrase pairs), where the other sub-models function together for providing the decoding information. This process is known as *decoding* in machine translation. The task is complicated since some target phrases may need movements. Generally speaking, an MT decoder extends a target translation left to right in the form of a set of hypotheses, each of which is a combination of one or more phrase translations. Figure 2.12 depicts part of the decoding process. The search begins with an empty hypothesis (no foreign words are translated). New hypotheses are created by extending the target output with phrase translations that cover some of the source words not yet translated. The joint probability of each new hypothesis is also computed according to the source words covered and the target words generated. Final states of the search are the hypotheses that cover all source words, among which the one with the highest probability is selected as the best target translation.

The above procedure describes the exhaustive search for machine translation. Theoretically, this approach is able to derive the best target translation when decoding with arbitrary phrase reorderings, however, it requires a factorial time complexity $O(C_{\bar{f}}I!)$. MT researchers have shown how using probabilistic models as well as restricting reordering conditions, to allow decoding within an acceptable time complexity. To the best of our knowledge, two decoders are well studied: one is the *beam search* decoder, first introduced by Tillmann (2001) and Och (2002) and then developed by Koehn (2004); the

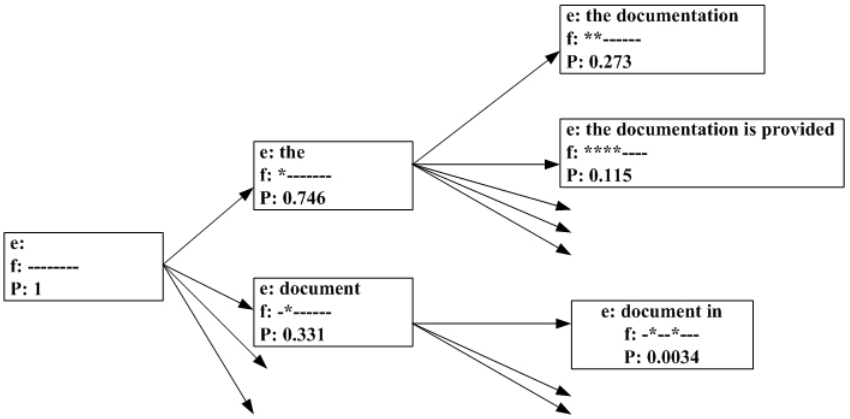


FIGURE 2.12: Exhaustive decoding process: state expansion. In each expansion, the target (English) words are generated (from left to right), the corresponding source (French) words are covered (marked by $*$), and the current probability score is computed. In this example, the input sentence is "*La documentation est incluse dans le kit d'assistance*" .

other is the *Cocke–Younger–Kasami* (CYK) style decoder, that is built from the *bracketing transduction grammar* (Wu, 1996). The rest of this section outlines the frameworks of both decoders and the readers are referred to (Koehn, 2004) and (Chiang, 2005) for detailed descriptions.

2.4.6.1 Beam search decoder

To reduce the time complexity to an acceptable level, the beam search decoder defines a set of stacks with a pre–fixed beam size, to which either a histogram pruning (keep the top B hypotheses and prune out others) or a threshold pruning (prune out hypotheses whose scores are lower than a threshold) is applied. Figure 2.13 shows part of the beam search process. When extending a new hypothesis, the decoder checks the corresponding stack and puts in the hypothesis if the stack is not full; otherwise it prunes out the inferior hypotheses that fall outside the beam.

The pseudo-code of the beam search decoder is given in Table 2.3. Suppose we use the histogram pruning with beam size B. In addition, let us define the length of the source sentence $\mathbf{f}$ as $|\mathbf{f}|$ and each source phrase $\bar{f}_i$ has at most $C_{\mathbf{f}}$ translation candidates, then the worst case time is $O(|\mathbf{f}|BC_{\mathbf{f}})$. This time complexity is linear because the beam search algorithm only expands B hypotheses at each level; it does not branch out more widely like many search algorithms that have exponential time complexities. The speed with which it executes is one of its greatest strengths and sustains its popularity in machine translation.

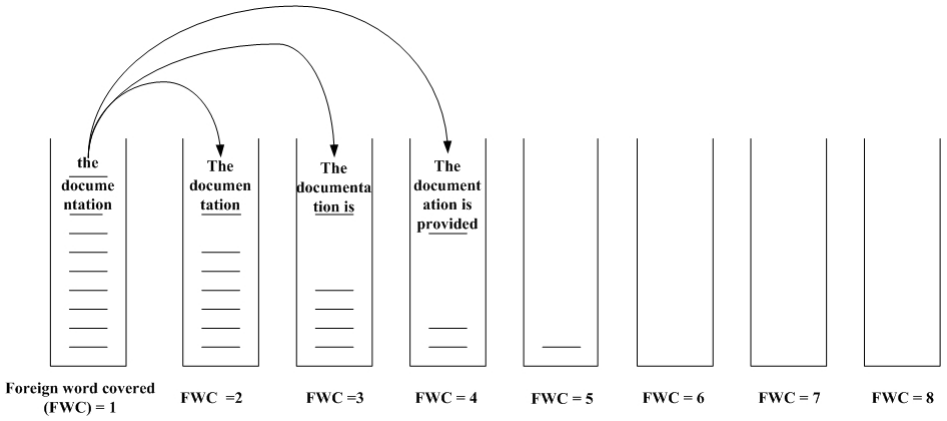


FIGURE 2.13: The beam search process: hypotheses are placed in stacks according to the number of source words covered so far. If a hypothesis is expanded into a new hypothesis (extended with a phrase option), it is placed in the corresponding stack. If the stack is full, the decoder would prune the inferior hypotheses outside the beam. In this example the input sentence is " *La documentation est incluse dans le kit d'assistance*".

Goal: Output a target translation **e**.

Input: the Start Hypotheses each of which consists of one phrase pair

1) **Initialise** $BEAM = \{\text{Start Hypotheses}\}$;
2) **repeat**
   $SET = BEAM$;
   $BEAM = \{\}$;
   for (each *state* in SET) **do**
     for (each *successor* of *state*) **do**
       $NewHypothesis$ =*state* extends with *successor*
       if ($BEAM$ is not full) **do**
         $BEAM = BEAM \bigcup \{NewHypothesis\}$
       else if ($NewHypothesis$ is not the worst hypothesis in $BEAM$)
         prune the inferior hypotheses in $BEAM$
         $BEAM = BEAM \bigcup \{NewHypothesis\}$
   until (all foreign words **f** are covered)

Output: the best translation in $BEAM$

TABLE 2.3: Pseudo-code of the beam search decoder.

2.4.6.2 Cocke–Younger–Kasami (CYK) style decoder

The *Cocke–Younger–Kasami* (CYK) algorithm (also known as *CKY*) is an example of dynamic programming, which determines whether a string can be generated by a set of context–free grammars. The application has its origins in the recognition and parsing literature (Cocke and Schwartz, 1970; Kasami, 1965; Younger, 1967), while later researchers found that it is simple to extend this algorithm to decode sentences with a set of context blocks (Chiang, 2005; Xiong et al., 2006).

Goal: Output a target translation **e**.

Input: phrase table and the length of the source sentence $N = |\mathbf{f}|$.

1) **Initialise** $H[N, N, B]$ as a matrix with $H[i, j, b]$ stores phrase $\bar{f} = [f_i, \ldots, f_j]$ and translation candidate $\bar{e}_b$;
2) **for** $i = 1$ to N **do**
  for $j = i + 1$ to N **do**
    for each state $H[i, j, b_{ij}]$ **do**
      for $k = j + 1$ to N **do**
        for each state $H[j, k, b_{jk}]$ do
          combine state $H[i, j, b_{ij}]$ and state $H[j, k, b_{jk}]$
          if combine successfully **do**
            if $H[i, k]$ is not full **do**
              add the new state in $H[i, k]$
            else
              add the new state and prune the inferior ones

Output: the best translation in $H[1, N]$

TABLE 2.4: Pseudo-code of the Cocke–Younger–Kasami style decoder.

The CYK style decoder follows a bottom up decoding order, which begins with the source phrases in each position. Table 2.4 demonstrates the pseudo-code of the decoding algorithm. In informal terms, translations for the phrase that covers source position i to j are placed in a hypothesis stack $H[i, j]$ (see Figure 2.14). Then for each hypothesis b_{ij} in $H[i, j]$ the decoder searches all possible extensions in the ending position k and add the new hypothesis in hypothesis stack $H[i, k]$. After the decoder travels through scenario in terms of the whole grid, the translations for the source sentence are placed in $H[1, N]$, where the best translation is picked up.

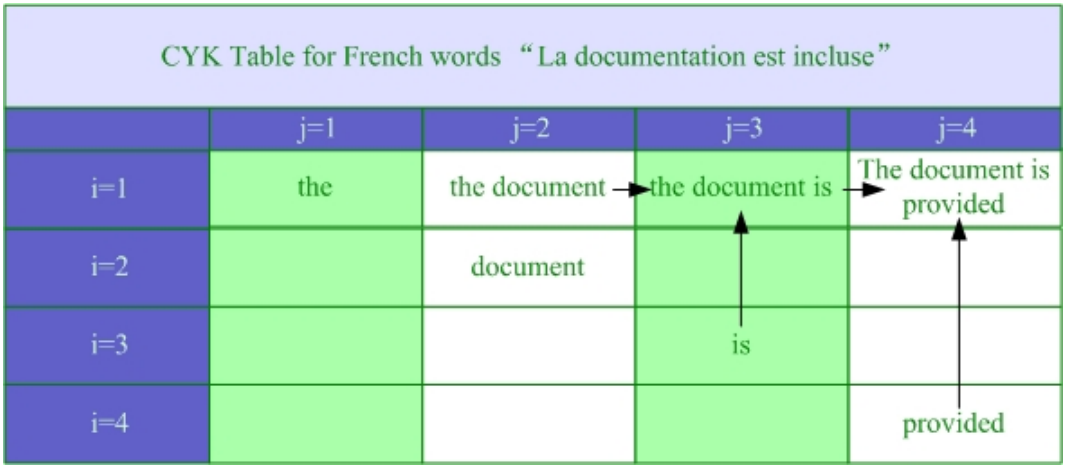


FIGURE 2.14: The CYK decoding process: hypotheses are placed in stacks according to the start and the end positions of the source phrases. If a hypothesis is expanded into a new hypothesis (extended with a consecutive phrase), it is placed in the corresponding stack. If the stack is full, the decoder would prune the inferior hypotheses outside the beam. In this example the input sentence is " *La documentation est incluse*".

If we restrict the beam size of each stack as B, then the worst case time of a CYK style decoder is $O(BN^3)$. This polynomial time complexity usually limits its application, especially when the source sentences are long.

2.4.7 Summary

In this section, we reviewed the sub-models composing an SMT system and the techniques applied. All models have already gone beyond the pure statistical methods, where various ML methodologies are introduced to boost the performance of translation. However, more powerful patterns in characterising the translation process and more robust learning agents are still needed, making exploiting and developing the existing models a worthful research task. In the next section, several MT systems utilising the sentence syntax information and cutting-edge ML approaches are introduced, enriching the patterns for machine translation.

2.5 Other MT systems

Apart from the SMT systems that make use of a combination of feature functions, discriminative structural methods have also led to significant advances in the MT field. The general procedure is to parse both source and target sentences into syntactic trees and learn a tree–to–tree transduction (e.g. Figure 2.15) before generating the translations. So far, studies of syntactic tree prediction for MT have produced improved results over some SMT systems and created their own MT faction. Owing to our research interest, we do not investigate syntactic tree prediction. However, the ML methods used in these systems still benefit our work and hence we outline three systems and discuss the ML methods adopted.

2.5.1 Syntactic tree prediction with the perceptron algorithm

The first approach is to train a discriminative model for the structured prediction and use a generative model to generate target translations. For example, (Collins and Roark, 2004) constructed a linear score model between source clauses $\mathbf{x}$ and target syntactic trees $\mathbf{y} = \langle d_1, \ldots, d_n \rangle$:

$$Score(\mathbf{x}, \mathbf{y}) = \sum_{j=1}^{n} \mathbf{w}^T \phi(\mathbf{x}, d_1, \ldots, d_j) \tag{2.17}$$

where ϕ are $\langle$source clause , target structure$\rangle$ features and $\mathbf{w}$ is the weight vector. Then they used the perceptron algorithm to train $\mathbf{w}$ and searched for the solution $\mathbf{y}^* = \arg\max_{\hat{\mathbf{y}}} Score(\mathbf{x}, \hat{\mathbf{y}})$ by beam search.

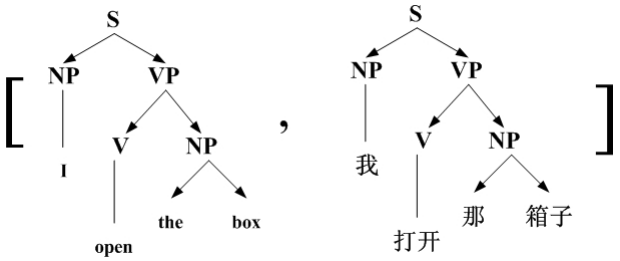


FIGURE 2.15: Example: the partial bitree used in (Turian et al., 2007). Each bitree contains one source parse tree and the corresponding target parse tree.

The proposed method is closely related to the PTP model used in an SMT system, while the phrase pairs are replaced by the ⟨source clause , target structure⟩ pairs and the translation probability (2.5) is changed to the score function (2.17). Although the syntactic trees utilised can provide syntax information for re-organising the target translations, the preliminary results in (Collins and Roark, 2004) have shown that this method lacks a language model, in which case the decoder finds it difficult to combine the target trees in a fluent way.

2.5.2 Syntactic tree prediction with the boosting technique

The second method was recently proposed by Turian et al. (2007), who treated the tree-to-tree transduction as a classification problem and optimised an objective function using a boosting technique. The proposed system parses both source and target sentences to construct multi–class classification decisions $\mathbf{X}$ (partial bitrees, see Figure 2.15), which make up the training sample pool. Each example i is expressed as a linear hypothesis

$$h_{\mathbf{w}}(i) = \mathbf{w}^T \mathbf{X}(i) = \sum_f w_f X_f(i) \tag{2.18}$$

where f ranges over all possible parse tree fragments. Then the boosting algorithm is employed to adjust $\mathbf{w}$ so as to minimise the log–loss objective function

$$J_{\mathbf{w}}(I) = \sum_{i \in I} \ln(1 + \exp\{-y_i \cdot h_{\mathbf{w}}(i)\}) + \lambda \sum_f |w_f|$$

During the decoding, the system parses the source test sentences and uses (2.18) to predict the target trees, which are then combined to generate the target translations.

In their pilot experiments, this method outperformed the SMT system – Pharaoh (Koehn, 2004) – with both BLEU evaluation and F1 measure. However, due to the syntactic tree representation it uses, the size of the sample pool is exponential in the number of training examples. As an example their experiments showed that for $10k$ training sentence pairs there were $819k$ bitrees and for $100k$ training sentence pairs there were $36.8M$ bitrees.

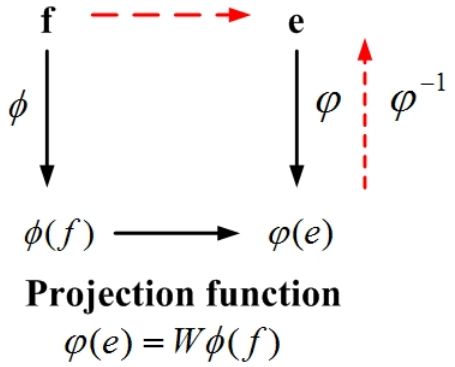


FIGURE 2.16: Illustration of the string-to-string prediction. Both source (**f**) and target (**e**) strings are embedded into their own feature spaces. Then a linear operator **W** is learnt from input to output.

In this case, the ever-increasing training time can negate the advantage of achieved accuracy.

2.5.3 Kernel–based methods for machine translation

The application of kernel–based methods to machine translation is much less explored. The primitive form can date back to the work of (Cortes et al., 2005), who formulated the relationship between two strings (**f** and **e**) with a string-to-string prediction framework (see Figure 2.16)

$$\varphi(\mathbf{e}) = \mathbf{W}\phi(\mathbf{f}) \tag{2.19}$$

Given a set of string pairs $\{\mathbf{f}^i, \mathbf{e}^i\}_{i=1}^N$, the method used ridge regression to learn the parameters **W**

$$\arg\min_{\mathbf{W}} J(\mathbf{W}) = \sum_{i=1}^{N} \|\mathbf{W}\phi(\mathbf{f}^i) - \varphi(\mathbf{e}^i)\|_{\mathcal{F}}^2 + \gamma\|\mathbf{W}\|_{\mathcal{F}}^2 \tag{2.20}$$

where $\|\mathbf{W}\|_{\mathcal{F}}^2$ denotes the Frobenius norm of the matrix-represented operator **W**

$$\|\mathbf{W}\|_{\mathcal{F}}^2 = \mathbf{tr}(\mathbf{W}^T\mathbf{W})$$

with **tr** denoting the trace operation.

Since the dimension of feature space is much larger than the number of training examples, Cortes et al. (2005) changed the problem to its dual representation and applied kernel ridge regression (described in Section 3.3.2) to derive the solution. Finally the prediction $\varphi(\mathbf{e})$ was passed through a *De Bruijn graph* (Kontorovich, 2004) to retrieve the target string **e**. Although this model has been used successfully in a variety of prediction tasks such as text and speech processing, two major limitations restrict its potential application to machine translation:

- Ridge regression requires a time complexity $O(dim(\phi)^3)$ for solving (2.20). This solution is impractical for MT tasks, where the dimension of linguistic feature space is considerably large. When changing to its dual representation, the dependency on the dimension is removed. However, the space complexity of kernel ridge regression is $O(N^2)$. For an MT corpus, which might contain hundreds of thousands of sentences, this space requirement is too large to handle.
- The decoder requires the strings to come from a regular language (e.g. finite state machine, turing machine, etc.), which cannot be proved true for human languages.

Subject to these restrictions, only a string–to–string prediction experiment (handwriting recognition) was carried out in (Cortes et al., 2005) , which is much simpler than an MT task.

Adopting this framework, Wang et al. (2007) utilised the least square approximation to derive $\mathbf{W}$, which ignores the regularizer $\gamma\|\mathbf{W}\|_{\mathcal{F}}^2$ in equation (2.20). In order to meet the requirements of an MT task, they replaced the De Bruijn graph with a beam search decoder and also made use of a language model to facilitate the fluency of translations. The resulting system is equivalent to an SMT system, where the PTP prediction is replaced by the string-to-string prediction (2.19). Although some improvements over a traditional SMT system (Koehn, 2004) have been reported, the application of this system is very limited. Due to its kernel–based score representation, the phrase reorderings in the system are very restricted. Moreover, the memory restriction pointed out above prevents its application to large-scale corpora, in which case only small-scale MT experiments were carried out (Wang et al., 2007).

2.5.4 Summary

Apart from the statistical methods, a variety of ML technologies has seeped into the MT field. They bring better translations by exploring extensive context and syntax information. But meanwhile, they complicate the MT systems and cause other problems such as increase of memory and decrease of speed. Which ML methodologies are appropriate for MT? How to make use of these methods? The answers of these questions are valuable, leading to a challenge to both MT and ML researchers.

2.6 Evaluation techniques for machine translation

Human evaluations for machine translation are extensive but expensive. Although human evaluations of machine translation weigh many aspects of translation such as adequacy, fidelity and fluency, they cost far too much time to finish. This is a big problem because researchers need to know how their systems work as soon as possible so as to

correct their ideas. In this case, the development of automatic evaluations for MT is essential.

The automatic evaluation of MT outputs should respect the quality of translations. However, the quality of a translation is inevitably subjective, there is no objective or quantifiable measurement for "good" or "bad". Therefore, the task for any automatic evaluation is to assign a score of quality which correlates with humans' judgement of quality. If we approximate humans' judgements as their translations (namely *references*), then the measure of correlations between MT outputs and references comes out to be the equivalent task. In this section, we introduce four types of automatic evaluations that are commonly used in the MT field.

2.6.1 Word error rate

Word error rate (WER) as a performance measure has its origins in the speech recognition literature. It is derived from the Levenshtein distance (Levenshtein, 1966) that measures the minimum number of modifications required to change one string to another. The only difference is that the word error rate works on the word level while the Levenshtein distance works on the character level.

To the best of our knowledge, Tillmann et al. (1997) first introduced WER as a metric of the performance of an SMT system. The computation of WER between a set of MT outputs $\{s_i\}_{i=1}^N$ and the corresponding references $\{z_i\}_{i=1}^N$ is expressed as

$$WER(\{s_i\}_{i=1}^N, \{z_i\}_{i=1}^N) := \frac{S + D + I}{N_{Ref}} \tag{2.21}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions and N_{Ref} is the number of words in the reference. Because it is expensive to check word alignments between the source sentences and the MT outputs, making it inconvenient to detect substitutions, substitutions are usually replaced by insertions and deletions (Tillmann et al., 1997).

A weak point of WER is the fact that the word orders are not taken into account appropriately. To overcome this problem, better measures that also consider phrase matches are in demand. As shown in (Callison-Burch et al., 2007), WER had a relatively low correlation with humans' evaluations, but its relationship to the sentence content still makes it a proper metric for evaluating machine translations.

2.6.2 BLEU score

Papineni et al. (2002) introduced another evaluation called BLEU, which takes the adequacy and fluency of translations into account. This evaluation was one of the first

evaluation to report high correlation with humans' judgement of quality and became one of the most popular evaluations in the MT field.

The criterion computes the geometric mean of the n–gram precisions of various length (generally up to 4) between a set of MT outputs $\{s_i\}_{i=1}^{N}$ and the corresponding references $\{z_i\}_{i=1}^{N}$

$$\text{BLEU} = \text{BP} \cdot \exp\big(\sum_{n=1}^{4} \frac{1}{4} \log p_n\big) \tag{2.22}$$

where $\text{BP} = \frac{1}{N} \sum_{i=1}^{N} \max\big(1, \exp(1 - |z_i|/|s_i|)\big)$ is the brevity penalty proportional to the average ratio of z_i/s_i lengths, and p_n is the average n–gram precision for the entire translation corpus

$$p_n = \frac{\sum_{i=1}^{N} \sum_{|u|=n} |\{u \mid u \in s_i \ \wedge \ u \in z_i\}|}{\sum_{i=1}^{N} \sum_{|u|=n} |\{u \mid u \in s_i\}|}$$

Unlike WER that concentrates only on words, BLEU uses a modified form of precision that also considers sequences of words (n–grams). Therefore, this method is able to evaluate the translation intelligibility or grammatical correctness between MT outputs and human references.

Although BLEU has significant advantages, there is no guarantee that an increase in the BLEU score is an indicator of improved translation quality. Indeed, one of the underlying assumptions of BLEU is that quality equals similarity to human translations. But this may be one reason for criticism as changing one word may result in different meaning which would not affect the BLEU score. For example, changing an MT output from "I do like football" to "I do not like football" (suppose the human reference is the former) only decreases BLEU a bit (when N is reasonable large), but the meaning of the translation has been totally changed.

2.6.3 NIST score

NIST is a method for evaluating the quality of MT outputs based on the BLEU score. Although it is derived from the BLEU evaluation criterion, it differs in one fundamental aspect: instead of n-gram precision, the information gain from each n-gram is taken into account. Mathematically, NIST is also a geometric mean of the form

$$\text{NIST} = \text{BP} \cdot \exp\big(\sum_{n=1}^{4} \frac{1}{4} \log p_n\big) \tag{2.23}$$

with $\text{BP} = \frac{1}{N}\sum_{i=1}^{N} \max\big(1, \exp(1-|z_i|/|s_i|)\big)$. But it replaces the average n–gram precision with

$$p_n = \frac{\sum_{i=1}^{N}\sum_{|u|=n} w_u |\{u \mid u \in s_i \ \wedge \ u \in z_i\}|}{\sum_{i=1}^{N}\sum_{|u|=n} |\{u \mid u \in s_i\}|}$$

where w_u is a pre-defined weight for each n–gram u. In informal terms, the rarer that n–gram u is, the larger w_u it receives.

The idea behind NIST is to give more credit if a system gives an n-gram match that is difficult, but to give less credit for an easy n-gram match. In other words, it emphasises more on the content of the sentence but less on the grammatical structure of the language.

2.6.4 METEOR score

The main principle behind the BLEU score and its derived NIST score is the measurement of the overlapping n-grams between an MT output and the reference, on the basis of the n–gram precision: the proportion of the matched n–grams out of the total number of n–grams in the MT outputs. The n–gram recall, the proportion of the matched n–grams out of the total number of n–grams in the references, is compensated using a fixed *brevity penalty* (BP). Alternatively Banerjee and Lavie (2005) argued that the n–gram recall is extremely important for assessing the quality of MT outputs, as it reflects to what degree the translation covers the entire content of an reference sentence. To sustain this argument, the METEOR evaluation was proposed.

Roughly speaking, the METEOR evaluates the harmonic–mean on explicit word–to–word matches between an MT output and its reference. The evaluation is divided into three steps

1. Create an alignment as a mapping of uni-grams between the MT output and the reference. Each uni-gram alignment indicates a correct word–to–word match.

2. Compute the uni-gram precision (P) and uni-gram recall (R) and derive the resulting F-mean:
$$Fmean = \frac{10PR}{R + 9P}$$

3. To take longer n-grams into account, METEOR computes a *penalty* to favour longer consecutive uni-gram matches. Then the final score is given by the formula
$$\text{METEOR} = Fmean \times (1 - penalty)$$

The readers are referred to (Banerjee and Lavie, 2005) for the detailed computation of the METEOR score. Compared with BLEU and NIST, METEOR concentrates more on evaluating the adequacy of MT outputs.

2.6.5 Optimising evaluation metrics for machine translation

Since the automatic evaluations of MT outputs are formulable, one can design different loss functions for evaluation metrics so as to optimise MT evaluations. For example, the loss function used in Chapter 4 aims at optimising the WER and the BLUE metrics implicitly, by means of optimising the phrase (n-gram) precisions. A method explicitly optimising the BLUE is discussed in (Wang and Shawe-Taylor, 2009). The loss function used in Chapter 5 optimises WER directly, which is similar to that of (Vickrey et al., 2005; Carpuat and Wu, 2007; Giménez and Màrquez, 2007). However, all of these methods do not optimise the evaluation metrics directly, because MT outputs are not generated directly by these models (see Chapter 4 and 5 for details). In the MT literature, directly optimising evaluation metrics is an active research area and currently a popular method utilised is (Och, 2003).

2.7 Machine translation systems: now and future

From its origins to present, the problem of learning language translation has experienced a systematic development. The early rule–based approaches attempt to generate the translations by applying a series of manually collected linguistic rules. The current SMT (or EBMT) systems make use of traditional statistical methods (e.g. maximum likelihood estimation) to mimic the cognitive process of human translators, for the purpose of automating the translation process. Lately, the development enters a new stage of collaboration between ML technologies and MT systems: the maximum entropy framework (Berger et al., 1996; Koehn et al., 2005; Zens and Ney, 2006) uses a set of probabilistic models for word (or phrase) translation and builds up the target translations to maximise a score function (a product of probabilities); discriminative structural methods (Collins and Roark, 2004; Turian et al., 2007) employ structured trees – syntactic trees of the source and the target phrases – to represent a hypothesis and predict the target translations on the basis of the classification techniques; kernel–based machine transductions (Cortes et al., 2005; Wang et al., 2007) project both input and output strings into a high dimensional feature space and utilise the regression techniques to learn a mapping from input to output.

There are two practical criteria to evaluate an MT system: the *quality of translation* and the *translation speed*. The rule–based approaches usually have good translation quality although their translation speed is very low, because they have to collect and revise the

linguistic rules manually. The statistical methods of SMT (and EBMT) systems are shown to have high translation speed, but they only consider the frequency features, which are too simple to solve problems met in the translation process (e.g. word sense disambiguation problem). The comprehensive pattern recognition techniques in ML are able to explore a vast amount of linguistic features that pure statistical methods do not have, and their ability to impose regularisations in a function space constraining the geometric structure brings in advantages that a pure statistical view does not bring. Unfortunately, they also cause new problems such as increase of memory and decrease of speed. All technologies have their own advantages and weaknesses, and many progresses are still required to pave the way for the success of machine translation.

Statistical machine translation has become a very active research field and has spawned a variety of systems that achieve state-of-the-art performance in some language pairs. Nevertheless, its sub-models are still to be improved. As a conclusion, we close this chapter with five open issues:

1. As discussed in Section 2.4.2, the traditional phrase translation probability (PTP) model only considers the frequency of aligned phrase pairs, which is not very reliable in phrase disambiguation. The potential solution is a better cooperation with ML techniques, which are able to take into account the sentence context where the phrase pairs come from as well as the potential connections between target translations.

2. It has been shown in Section 2.4.4 that the discriminative phrase reordering models usually outperform the generative ones. However, even the discriminative models can not capture phrase movements perfectly, especially long distance reorderings. Moreover, little analysis is carried out for the performance of phrase movements, the causes of the improvement of translation quality are still unclear.

3. The current ML techniques applied to SMT systems are not satisfactory, they could be either too simple for the real tasks or time consuming. What should be applied, when and where they are better to apply is a question needed to be answered.

4. The performance of the PTP model mainly depends on the accuracy of word alignments it receives. Since the word alignments generated by the alignment toolkit are not 100% accurate and the current phrase extraction technique is heuristic, any creation in word alignment or modification in the phrase extraction procedure might help improve the quality of translation.

5. Further work is also envisaged to improve the decoding performance by a better score function (i.e. better MT framework), in terms of reordering constraints and future score estimation.

As a good compensation and potential replacement of pure statistical methods, the applications of ML technologies grow rapidly and become a new and emerging area of machine translation. The goal of this thesis is also to explore and develop ML methodologies for MT. In each application we create a new MT system or propose a specific sub-model for current SMT systems, by means of different cutting-edge ML approaches. To provide the readers a general view of the ML methods utilised, a systematic description of these methods is given in the next chapter.

Chapter 3

Machine learning: a new era for machine translation

Symbol	Notation
$\mathcal{S}$	a set of examples
$\mathbf{x}_i$	the input of the i-th example in $\mathcal{S}$
$\mathbf{X}$	the input matrix including all examples in $\mathcal{S}$ $\mathbf{X} = [\mathbf{x}_1^T, \ldots, \mathbf{x}_N^T]^T$
$\mathcal{X}$	the input attribute space $\mathbf{x}_i \in \mathcal{X}$
y_i	the output of the i-th example in $\mathcal{S}$
$\mathbf{y}$	the output matrix including all examples in $\mathcal{S}$ $\mathbf{y} = [y_1, \ldots, y_N]^T$
$\mathbf{r}_i$	the structured output of the i-th example in $\mathcal{S}$
$\mathbf{R}$	the structured output space $\mathbf{r}_i \in \mathbf{R}$
N	the number of examples in $\mathcal{S}$
$\phi(\mathbf{x})$	the feature vector of input $\mathbf{x}$
Φ	the feature matrix including all examples in $\mathcal{S}$ $\Phi = [\phi(\mathbf{x}_1)^T, \ldots, \phi(\mathbf{x}_N)^T]^T$
$\mathcal{H}_\phi$	the input feature space $\phi(\mathbf{x}) \in \mathcal{H}_\phi$
$\varphi(y)$	the feature vector of output y
$\mathcal{H}_\varphi$	the output feature space $\varphi(y) \in \mathcal{H}_\varphi$
d	the dimension of feature space
dim	the dimension of
$\mathbf{w}$	a weight vector $\mathbf{w} \in \mathbb{R}^d$
$\mathbf{W}$	a matrix-represented weight operator $\mathbf{W} \in \mathbb{R}^{dim(\varphi) \times dim(\phi)}$
$f(\mathbf{x})$	a linear evaluation function $f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$
$J(\mathbf{w})$	a risk function with respect to $\mathbf{w}$ for an optimisation problem
$\rho(y, f(\mathbf{x}))$	a function measuring the loss between $f(\mathbf{x})$ and the true output y

TABLE 3.1: Notations used in this chapter.

3.1 Introduction

This chapter reviews the algorithmic aspects of machine learning used to formulate and solve machine translation problems described later in the thesis. We start with some basic aspects of formulating supervised learning problems as large margin classification problems. A particular aspect of applying machine learning to natural language processing (NLP) tasks is the large number of output classes. We review multi-class classification techniques and discuss a recently developed, particularly novel method, known as *maximum margin regression* (MMR). The main novelty of the work presented in this thesis is taking advantage of structure in the output space by the formulation of structured learning problems. We discuss structured learning problems in Section 3.4.3 and review how other authors have taken advantage of output structure. A perceptron–based algorithm due to (Collins, 2002) forms the basis of the training the models formulated in this thesis. Collins' algorithm and its properties are discussed in Section 3.4.3.6. A family of other related algorithms are then reviewed in Section 3.5.

3.2 Basics of supervised learning

In supervised learning the problem is formulated in terms of pairs of data items consisting of inputs and outputs. When the outputs are continuous values the problem is referred to as a *regression* problem, while when the outputs are labels indicating class memberships, we refer to *classification* problems.

We will mainly discuss classification problems because that is how NLP tasks are formulated in this study. In particular, we restrict our attention to *linear discriminant functions*, namely those for which the decision surfaces are hyperplanes.

Definition 3.1. (**Linear discriminant function**) A *linear discriminant function* is obtained by taking a linear function of the *input attributes* $\mathbf{x}$ such that

$$f(\mathbf{x}) := \mathbf{w}^T\mathbf{x} + w_0 \tag{3.1}$$

where $(\mathbf{w}, w_0)$ denotes a weight vector that maps the input to the output.

To either classify a new example or predict its value, it is necessary to know the evaluation function $f(\mathbf{x})$. There has been a variety of learning models, utilising Bayesian theory, convex loss optimisation, boosting and so forth. Amongst these methods one fundamental learning framework originates from a simple probabilistic model – the *Gaussian noise model*, where more and more assumptions (prior knowledge) are added and it breeds a big family of supervised learning.

3.2.1 Gaussian noise model and least square approximation

Assume that the output y is given by a deterministic function $f(\mathbf{x})$ with additive Gaussian noise such that

$$y = f(\mathbf{x}) + \epsilon \tag{3.2}$$

where ϵ is a zero mean Gaussian random variable with variance σ^2. Then the conditional probability of the output y given $\mathbf{x}$ is of the form

$$p(y|\mathbf{x}, f, \sigma^2) = \mathcal{N}\big(y|f(\mathbf{x}), \sigma^2\big) \tag{3.3}$$

Now consider a training set $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$ sampling *independent and identically–distributed* (i.i.d) from the distribution (3.3), the following expression holds for the likelihood of observing $\mathcal{S}$

$$P(\mathbf{y}|\mathbf{X}, f, \sigma^2) = \prod_{i=1}^{N} \mathcal{N}\big(y_i|f(\mathbf{x}_i), \sigma^2\big) \tag{3.4}$$

Making use of the standard form for the Gaussian distribution

$$\mathcal{N}\big(y|f(\mathbf{x}), \sigma^2\big) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\Big\{ - \frac{\big(y - f(\mathbf{x})\big)^2}{2\sigma^2} \Big\} \tag{3.5}$$

and taking the logarithm of the likelihood function, we have

$$\ln P(\mathbf{y}|\mathbf{X}, f, \sigma^2) = \; -\tfrac{N}{2} \ln \sigma^2 - \tfrac{N}{2} \ln(2\pi) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \big(y_i - f(\mathbf{x}_i)\big)^2 \tag{3.6}$$

Clearly maximising this likelihood is equivalent to minimising the following risk function

$$J(\mathcal{S}|f, \sigma^2) = \frac{1}{2\sigma^2} \sum_{i=1}^{N} \big(y_i - f(\mathbf{x}_i)\big)^2 + \frac{N}{2} \ln \sigma^2 \tag{3.7}$$

Let σ^2 be one parameter, we can then use *maximum likelihood estimation* (MLE) to determine the variance of the Gaussian conditional distribution. Minimising (3.7) with respect to σ^2 gives

$$\frac{1}{\sigma^2} = \frac{1}{N} \sum_{i=1}^{N} \big(y_i - f(\mathbf{x}_i)\big)^2 \tag{3.8}$$

Since the variance σ^2 totally depends on the evaluation function $f(\mathbf{x})$, we can first estimate $f(\mathbf{x})$ by minimising a simplified risk function

$$J(\mathcal{S}|f) = \sum_{i=1}^{N} \big(y_i - f(\mathbf{x}_i)\big)^2 \tag{3.9}$$

and subsequently use this solution to find the variance σ^2.

3.2.1.1 least square approximation

In the view of supervised learning, model (3.9) attempts to find an evaluation function $f(\mathbf{x})$ for which the sum of the squares of the training errors is minimised. This well–studied model is the most commonly chosen measure of the collective discrepancy between the training data and the evaluation function, which is also known as *least square approximation (LSA)* introduced by Gauss (Stigler, 1986). Compared with other models, its simple analytic solution is one of its strengths.

First let us consider the linear models without bias only $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. Let $\mathbf{X} = [\mathbf{x}_1^T, \ldots, \mathbf{x}_N^T]^T$ be the input matrix and $\mathbf{y} = [y_1, \ldots, y_N]^T$ be the output vector, then the optimal solution of LSA is able to be written analytically, which is given by the formula

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{3.10}$$

To consider the bias term w_0, the input vector can be extended with one constant $\bar{\mathbf{x}} = [\mathbf{x}^T, 1]^T$ and set $\bar{\mathbf{X}} = [\bar{\mathbf{x}}_0^T, \ldots, \bar{\mathbf{x}}_N^T]^T$, then the solution of $f(\mathbf{x})$ with a bias term is $\bar{\mathbf{w}} = (\bar{\mathbf{X}}^T\bar{\mathbf{X}})^{-1}\bar{\mathbf{X}}^T\mathbf{y}$ with $\bar{\mathbf{w}} = [\mathbf{w}^T, w_0]^T$.

The prediction of a new example $\mathbf{x}$ can then be computed using the evaluation function $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$. Since the bias term w_0 can always be represented by extending the input vector with one constant, the notation of a linear discriminant function is usually simplified as $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ without loss of clarity.

To minimise the squared loss of a linear discriminant function, one needs to maintain as many parameters as dimensions d and solve a $(d+1) \times (d+1)$ system of linear equations that requires a computational complexity $O(d^3)$.

3.2.2 Over-fitting and regularisation

Although having a simple analytic solution, the LSA method is usually prone to *over–fitting* (Bishop, 2006). An overfitted model is a model that contains more unknown parameters than can be justified by the data (Everitt, 1998). In other words, the evaluation function $f(\mathbf{x})$ has too much freedom to perfectly fit to the training data, while giving a very poor representation of the data distribution.

One technique that is often used to control the over-fitting phenomenon in such case is that of *regularisation*, which involves adding a penalty term to the risk function (3.9) so as to restrict the freedom of $f(\mathbf{x})$. The simplest way is to take the norm of the weight vector as a penalty, leading to a modified risk function of the form

$$J(\mathcal{S}|\mathbf{w}) = \sum_{i=1}^{N} \left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{3.11}$$

where the scaling factor λ governs the relative importance of the regulariser.

There is a Bayesian interpretation for formulation (3.11), namely *maximum a posteriori* (MAP) estimation, which imposes a prior distribution of weight parameters for the Gaussian noise model and maximise the posterior of f by means of *Bayesian inference*. We will not enter into a Bayesian discussion in this thesis and the readers are referred to (Bishop, 2006) for details.

The solution of (3.11) is well-known as *ridge regression* (Hoerl and Kennard, 1970), which is computed by

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda I_d)^{-1}\mathbf{X}^T\mathbf{y}. \tag{3.12}$$

Identical to LSA, the ridge regression solution has to maintain d (or $d+1$) parameters and requires a computational complexity $O(d^3)$ for solving the linear equations.

Regularisation allows complex models to be trained on data sets of limited size without severe over-fitting. However, the problem is then shifted to the one of determining a suitable value of the scaling factor λ. In practice, this problem is usually solved by a technique called *cross-validation* (Kohavi, 1995). That is, the data are randomly split into the training and the validation sets; the model with a pre-defined λ is then fit to the training data, while predictive accuracy is assessed using the validation set. The average accuracy over the splits is then taken as the predictive performance of this λ. The process goes through a set of λ, from which the one with the highest predictive performance is selected.

3.2.3 A generalisation of supervised learning – risk function $J(\mathbf{w})$

One important observation from previous examples is that both optimisations arrive at minimising the same risk function with variant formats. From a machine learning perspective, this observation can be generalised to a general form of risk function

$$J(f) := \mathbb{E}_{emp}[f] + \lambda\Lambda[f] \tag{3.13}$$

where

$$\mathbb{E}_{emp}[f] := \frac{1}{N}\sum_{i=1}^{N} \rho(y_i, f(\mathbf{x}_i)) \tag{3.14}$$

denotes the *empirical error* and $\Lambda[f]$ is a *regulariser* to prevent over-fitting.

With different settings of $\Lambda[f]$, equation (3.13) can spawn a number of Bayesian models such as MLE and MAP. Furthermore, by choosing different types of loss function $\rho(y, f(\mathbf{x}))$ it arrives at comprehensive discriminative models, including least square approximation, ridge regression and support vector machines.

3.3 Embedding technologies – from feature space to kernel methods

One of the most predominant problems in machine learning is the application of various methods to real world data. Among these methods, the linear discriminant function (3.1) is popular because of its good analytic properties. However, the real world data could be highly non-linear such that distinguishing the classes (or predicting the values) in a linear fashion is infeasible to achieve. Although this can be overcome by transforming the original input space into a higher dimensional feature space, the new feature embedding may become computationally expensive to perform. Fortunately, by applying the commonly known "kernel trick", it is possible to embed the data into an appropriate feature space while preserving the computational complexity.

3.3.1 Formulating classification problems

In the field of supervised learning, *classification* is a procedure to place individual items into groups, on the basis of quantitative information on one or more characteristics (*input attributes*) of the items. One typical classifier is the *linear binary classifier*. Mathematically, given a training set $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, the classifier is of the form $f(\mathbf{x}_i) = \mathbf{w}^T\mathbf{x}_i + w_0$, where $f(\mathbf{x}_i) > 0$ suggests that the input $\mathbf{x}_i$ is assigned to the positive class while $f(\mathbf{x}_i) < 0$ suggests the negative one. Figure 3.1 demonstrates a separation of two sets of objects "□" and "○". It is clear that the affine linear space $(\mathbf{w}, w_0)$ defines a hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$ that separates the attributes' space into two half spaces. These two half spaces then correspond respectively to the data inputs of the two distinct classes.

Definition 3.2. (**Hyperplane**) A *hyperplane* is an affine linear space of dimension $d - 1$, that divides the space into two half spaces.

Definition 3.3. (**Separating hyperplane**) A *separating hyperplane* is a hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$, satisfying

$$\begin{cases} \mathbf{w}^T\mathbf{x} + w_0 \geq 0 & if \quad y = 1 \\ \mathbf{w}^T\mathbf{x} + w_0 < 0 & if \quad y = -1 \end{cases} \tag{3.15}$$

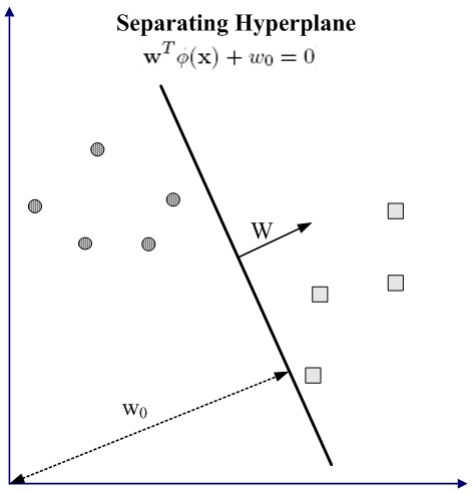


FIGURE 3.1: A separating hyperplane for a two dimensional data set.

The linear classifiers are severely limited as they can only apply to data that are linearly separable. However, the real word applications usually require more complex expressions which are non–linear in the data attributes.

3.3.2 Learning in a feature space and the dual theorem

The primary weakness of linear discriminant functions is that real world data may not be linearly separable in data attribute space. This suggests that if the linear models are to be powerful enough for the discrimination task, more abstract attributes of the data are needed. To meet this requirement, a pre–processing in machine learning – the *feature space projection* of the data is commonly used. In effect, it is equivalent to mapping the input space $\mathcal{X} \in \mathbb{R}^d$ into a higher dimensional space $\mathcal{H}_\phi = \{\phi(\mathbf{x}) \in \mathbb{R}^D : \mathbf{x} \in \mathcal{X}\}$ where the discriminant may benefit from the new representation. The resulting attributes derived are usually referred to as *features* and the corresponding discriminant function is expressed as $f(\mathbf{x}) := \mathbf{w}^T\phi(\mathbf{x})$.

However, one problem with this explicit feature representation is the increasing number of dimensions, which very quickly makes computation unpractical. Take the ridge regression (3.12) for example, the solution with a feature representation is of the form

$$\mathbf{w} = \left(\Phi^T\Phi + \lambda I_D\right)^{-1}\Phi^T\mathbf{y} \tag{3.16}$$

where $\Phi = [\phi(\mathbf{x}_1)^T, \ldots, \phi(\mathbf{x}_N)^T]^T$.

Consider a d dimensional input space $\mathcal{X} \in \mathbb{R}^d$ together with a feature projection

$$\phi : \mathbf{x} = (x_1, \ldots, x_d) \mapsto \phi(\mathbf{x}) = (x_i x_j)_{i,j=1}^d \in \mathcal{F} = \mathbb{R}^D$$

which has $D = d^2$ features, then the computational complexity of (3.16) would increase from $O(d^3)$ to $O(d^6)$. Such time increase is endurable for some data sets (e.g. the "school data"[1]), in which the number of input attributes is small. However, for others (e.g. gene strings, text) that might have hundreds of thousands of attributes, the computation with the explicit feature projection will quickly become unfeasible.

One approach to address this problem is to reduce the weight parameters from the number of features to the number of data samples, which involves the application of the *dual representation*. Again, take ridge regression for example, optimising risk function (3.11) is equivalent to solving the following dual optimisation problem

$$\begin{aligned} \max_{\boldsymbol{\alpha},\lambda} \quad & -\lambda^2 \boldsymbol{\alpha}^T \boldsymbol{\alpha} + 2\lambda \boldsymbol{\alpha}^T \mathbf{y} - \boldsymbol{\alpha}^T \Phi \Phi^T \boldsymbol{\alpha} - \lambda R^2 \\ s.t. \quad & \lambda \geq 0 \quad R \geq 0 \end{aligned} \tag{3.17}$$

where $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T$ is the so-called *dual variables*, whose analytic solution is given by the formula

$$\bar{\boldsymbol{\alpha}} = (\Phi \Phi^T + \lambda I_N)^{-1} \mathbf{y} \tag{3.18}$$

This dual form solution is also known as *kernel ridge regression* (Saunders et al., 1998), in which the prediction for a new sample $\mathbf{x}$ is computed by

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle \tag{3.19}$$

The breakthrough of using the dual representation is that the computational complexity of the solution becomes $O(N^3)$, which is a large reduction when $D \gg N$, although computing $\Phi \Phi^T$ still requires a time cost $O(N^2 D)$.

3.3.3 Kernel function and kernel matrix

By applying the dual representation, the computation in the feature space is reduced to two aspects: one is for the solution of dual parameters $\boldsymbol{\alpha}$ (3.18) and the other is for the prediction (3.19). This results in a time complexity $O(N^3 + N^2 D)$ that still depends on the dimension of feature space. Fortunately, this computation is possible to be further reduced via what is commonly known as the *kernel trick*, that allows us to compute the inner product in the feature space $\mathcal{H}_\phi$ without the explicit feature mapping ϕ.

[1] The data set is from the Inner London Education Authority and is available at `http://multilevel.ioe.ac.uk/intro/datasets.html`. It consists of examination records of 15,362 students from 139 secondary schools, each of which has 27 input attributes such as gender, ethnic group, school denomination, etc.

Definition 3.4. (**Kernel function**) A *kernel* is a function K that for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$ satisfies

$$K(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle \tag{3.20}$$

where ϕ is a feature projection from $\mathcal{X}$ to an (inner product) feature space $\mathcal{H}_\phi$

$$\phi : \mathbf{x} \in \mathbf{X} \in \mathbb{R}^d \mapsto \phi(\mathbf{x}) \in \mathcal{H}_\phi \in \mathbb{R}^D$$

.

If the kernel function applies to all pairs of data samples, this involves the construction of a kernel matrix.

Definition 3.5. (**Kernel matrix**) Given a kernel function K and inputs $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathcal{X}$, the $N \times N$ matrix

$$\mathbf{K}_{i,j} = K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \quad \forall i, j = 1, \ldots, N \tag{3.21}$$

is named *kernel matrix* $\mathbf{K}$ of K with respect to $\mathbf{x}_1, \ldots, \mathbf{x}_N$.

Note that the kernel matrix $\mathbf{K} = \Phi\Phi^T$. Substituting this representation into the solution (3.18), we have $\bar{\boldsymbol{\alpha}} = (\mathbf{K} + \lambda I_N)^{-1}\mathbf{y}$ and the prediction of a new example $\mathbf{x}$ is given by $f(\mathbf{x}) = \sum_i \alpha_i K(\mathbf{x}_i, \mathbf{x})$.

Incorporated with the kernel matrix, the computational complexity of the solution $\boldsymbol{\alpha}$ is able to reduce from $O(DN^2 + N^3)$ to $O(dN^2 + N^3)$ (Shawe-Taylor and Cristianini, 2004). In this case, the kernel functions are able to reduce the time for computing inner products, rendering the algorithms more efficient in very high–dimensional feature spaces[2].

The kernel functions have been introduced for sequence data, graphs, text, images as well as vectors. Throughout the thesis, the essential kernel functions used are from the family of string kernels, which will be specified in Chapter 4.

3.3.4 Summary

Started from data attributes, followed by designing a feature space and finally arrived at kernel functions, the development of feature space projection techniques allows non-linear approaches in the data attribute space to be achieved by linear approaches in a higher dimensional feature space, which is expected to provide more flexibility in data expression. At the same time, it complicates the data expression and requires more

[2]Since $\mathcal{H}_\phi$ can represent the original attribute space ($\phi(\mathbf{x}) = \mathbf{x}$) as well as a more complicated feature space, the feature representation $\phi(\mathbf{x})$ will be used instead of $\mathbf{x}$ throughout the thesis without loss of clarity. Meanwhile, the dimension of $\phi(\mathbf{x})$ is also written as d (or $d + 1$ if specified) instead of D.

efficient and robust ML methodologies that are better in exploring and exploiting the ever-increasing data characteristics.

3.4 Maximum margin classifier

Maximum margin classifier has been successfully applied to natural language processing and statistical machine translation (Tsochantaridis et al., 2004; Giménez and Màrquez, 2007). The consequences were sufficiently encouraging that an EU project – *Statistical Multilingual Analysis for Retrieval and Translation*[3] (SMART) has been setup, with the intention of developing new maximum margin learners for machine translation. Since our work is partly supported by the European Commission under the SMART project, this thesis also considers this classifier as the central optimisation model. In order to provide the readers the basics of this field, we state several traditional and state-of-the-art maximum margin classifiers in the rest of this section.

The previous sections comment on the linear classifiers that separate d dimensional examples with a $d-1$ dimensional hyperplane. In the case of maximum margin classifier, we are additionally interested in finding out whether we can achieve a maximum separation (margin) between two classes. If such a hyperplane exists, it is of our interest and is known as a *maximum–margin hyperplane*, and such a classifier is known as a *maximum margin classifier*.

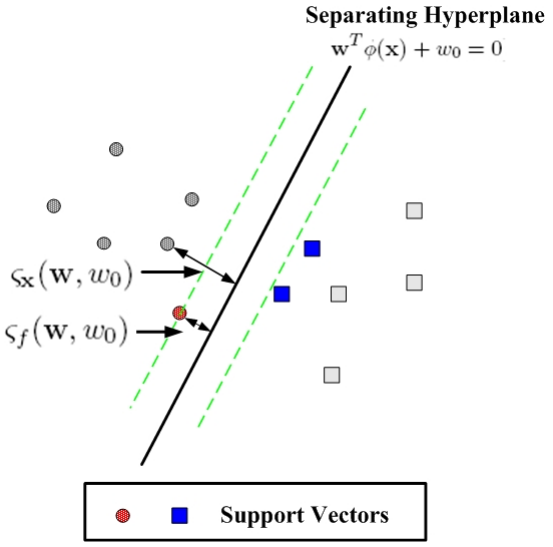


FIGURE 3.2: An example of the separating hyperplane and the margin.

Definition 3.6. (**Margin of a separating hyperplane**)

Let $\mathcal{S} = \{(\phi(\mathbf{x}_i), y_i)\}_{i=1}^N \in \mathbb{R}^{d+1} \times \{-1, 1\}$ denote the training set, and $f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + w_0$ is the corresponding discriminant function. If the data are linearly separable, there

[3] `http://www.smart-project.eu/node/1`

exists $\mathbf{w} \in \mathbb{R}^d$ and $w_0 \in \mathbb{R}$ such that

$$y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0) > 0, \quad i = 1, \ldots, N.$$

By denoting the distance $\varsigma_{\mathbf{x}}(\mathbf{w}, w_0)$ of a sample $\mathbf{x}$ from a hyperplane $f(\mathbf{x}) = 0$ as (see Figure 3.2)

$$\varsigma_{\mathbf{x}}(\mathbf{w}, w_0) = \frac{|\mathbf{w}^T\phi(\mathbf{x}) + w_0|}{\|\mathbf{w}\|} \tag{3.22}$$

the separating hyperplane of the training set S achieves a *margin*

$$\varsigma_f(\mathbf{w}, w_0) = \min_{i=1}^{N} \varsigma_{\mathbf{x}_i}(\mathbf{w}, w_0) \tag{3.23}$$

The maximum margin solution can be motivated by the *statistical learning theory* (Vapnik, 1995). Alternatively, we will show in Section 3.4.1.3 that by incorporating the Canonical hyperplane constraint, maximum margin is also a form of regularisation, that restricts the freedom of a discriminant function.

3.4.1 Binary optimal separating hyperplane – Support Vector Machine

A *Support Vector Machine* (SVM) (Cortes and Vapnik, 1995) is a binary maximum margin classifier, that learns a separating hyperplane with the maximum margin. Mathematically, it builds up the following optimisation problem

$$\begin{aligned} \max_{\mathbf{w}, w_0} \quad & \min_i \left\{ \frac{y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0\big)}{\|\mathbf{w}\|} \right\} \\ s.t. \quad & y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0\big) \geq 0 \quad i = 1, \ldots, N \end{aligned} \tag{3.24}$$

A separating hyperplane is parameterised by $(\mathbf{w}, w_0)$, but the choice is not unique as rescaling with a positive constant gives the same separating hyperplane. Hence, a data–dependent parametrisation, namely *Canonical hyperplane* (Vapnik, 1995), is usually applied to fix the optimal hyperplane, which requires

$$\min_{i=1}^{N} y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0\big) = 1 \tag{3.25}$$

Adding the Canonical hyperplane constraint to problem (3.24) yields the primal version of the *hard margin SVM* optimisation:

$$\begin{aligned} &\max_{\mathbf{w},w_0} \quad \frac{1}{\|\mathbf{w}\|} \\ &s.t. \quad y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i)+w_0\big) \geq 0 \quad i=1,\ldots,N \\ &\qquad \min_i y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i)+w_0\big) = 1 \\ \Rightarrow \\ &\min_{\mathbf{w},w_0} \quad \tfrac{1}{2}\mathbf{w}^T\mathbf{w} \\ &s.t. \quad y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i)+w_0\big) \geq 1 \quad i=1,\ldots,N \end{aligned} \tag{3.26}$$

where the word "hard" implies that no training errors are allowed (i.e. the data are linearly separable in the feature space).

3.4.1.1 Dual version of the hard margin SVM

By introducing a set of dual variables $\boldsymbol{\alpha}$ and applying the dual theorem, the problem (3.26) is shifted to its dual representation:

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} \quad -\tfrac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}_\mathbf{X}\mathbf{K}_\mathbf{y}\boldsymbol{\alpha} + \mathbf{1}^T\boldsymbol{\alpha} \\ &s.t. \quad \mathbf{y}^T\boldsymbol{\alpha} = 0 \\ &\qquad \boldsymbol{\alpha} = \{\alpha_i | \alpha_i \geq 0 \; i=1,\ldots,N\} \end{aligned} \tag{3.27}$$

where $\mathbf{1}$ denotes a vector with components 1 and $\mathbf{K}_\mathbf{X}$ and $\mathbf{K}_\mathbf{y}$ are kernel matrices

$$\mathbf{K}_\mathbf{X} = \{\langle\phi(\mathbf{x}_i),\phi(\mathbf{x}_j)\rangle : \; i,j=1,\ldots,N\}, \tag{3.28}$$

$$\mathbf{K}_\mathbf{y} = \{y_i y_j : \; i,j=1,\ldots,N\}. \tag{3.29}$$

If $\bar{\boldsymbol{\alpha}}$ is a solution of (3.27), the solution $\bar{\mathbf{w}}$ of (3.26) is then computed by

$$\bar{\mathbf{w}} = \sum_{i=1}^{N} \bar{\alpha}_i y_i \phi(\mathbf{x}_i). \tag{3.30}$$

Since $\bar{\mathbf{w}}$ is a linear combination of only the $\mathbf{x}_i$ for which $\bar{\alpha_i} > 0$, these $\mathbf{x}_i$ are termed *support vectors*.

In addition, the bias parameter w_0 can be determined by a support vector $\mathbf{x}_j$ using

$$w_0 = y_j - \bar{\mathbf{w}}^T\phi(\mathbf{x}_j) \tag{3.31}$$

Finally, the prediction of a new example $\mathbf{x}$ is given by

$$y := sgn(\mathbf{w}^T\phi(\mathbf{x}) + w_0) = sgn(\sum_{i=1}^{N} \alpha_i y_i \mathbf{K}_{\mathbf{X}}(\mathbf{x}_i, \mathbf{x}) + w_0) \tag{3.32}$$

with function $sgn(\bullet)$ gives the sign of the expression.

3.4.1.2 Linearly nonseparable case – the soft margin SVM

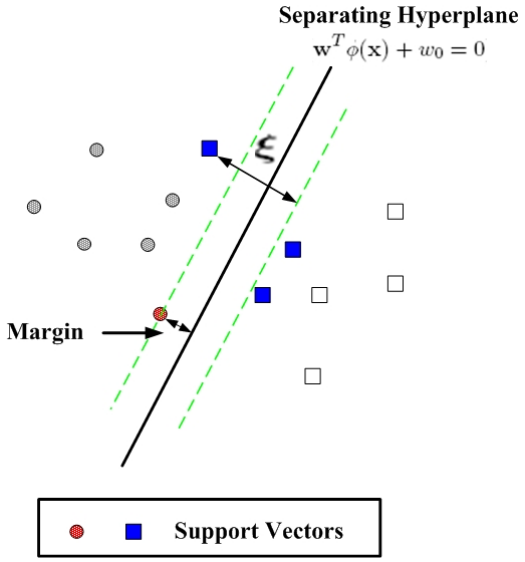


FIGURE 3.3: An example of a soft margin SVM classifier.

If the data are not linearly separable (Figure 3.3), there exist two schemes to overcome this problem. One is to transform the input space into an appropriate higher dimensional space in which the examples are linearly separable. This scheme resorts to creating an appropriate kernel (Shawe-Taylor and Cristianini, 2004), and several approaches have been proposed in the literature (Joachims et al., 2001; Gao et al., 2002; Argyriou et al., 2006). However, the applications of these methods are still limited and designing proper kernels for specific problems present a challenge to ML developers. The other scheme is to lose certain accuracy and relax the separation constraints with a slack variables ξ, which allows for non–separable data with a penalty proportional to the amount of examples that are misclassified (see Figure 3.3). In other words, an input $\mathbf{x}_i$ which cannot be separated correctly by the hyperplane would incur an empirical error ξ_i, then the goal is to find a separating hyperplane such that a maximum margin is achieved with minimum empirical errors. This modified optimisation problem is known as the

soft margin SVM optimisation (Cortes and Vapnik, 1995), which is of the form

$$\begin{array}{ll} \min\limits_{\mathbf{w},w_0,\boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\mathbf{1}^T\boldsymbol{\xi} \\ s.t. & y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0\big) \geq 1 - \xi_i \quad i = 1,\ldots,N \\ & \boldsymbol{\xi} := \{\xi_i | \xi_i \geq 0,\, i = 1,\ldots,N\} \end{array} \tag{3.33}$$

Note that the hard margin SVM is a special case of the soft margin SVM, with the assumption that the data are linearly separable (i.e. $C = \infty$).

Similar to that of the hard margin SVM, the dual representation of the soft margin SVM is expressed as

$$\begin{array}{ll} \max\limits_{\boldsymbol{\alpha}} & -\frac{1}{2}\boldsymbol{\alpha}^T\mathbf{K}_\mathbf{X}\mathbf{K}_\mathbf{y}\boldsymbol{\alpha} + \mathbf{1}^T\boldsymbol{\alpha} \\ s.t. & \mathbf{y}^T\boldsymbol{\alpha} = 0 \\ & \boldsymbol{\alpha} = \{\alpha_i | 0 \leq \alpha_i \leq C \; i = 1,\ldots,N\} \end{array} \tag{3.34}$$

which contains one equation constraint and N box constraints.

Again examples for which the optimal solution $\bar{\alpha}_i > 0$ are termed support vectors and the prime solution $\bar{\mathbf{w}}$ is given by $\bar{\mathbf{w}} = \sum_{i=1}^{N} \bar{\alpha}_i y_i \phi(\mathbf{x}_i)$. Meanwhile, the bias term $\bar{w}_0$ can be determined by the Karush–Kuhn–Tucker conditions (Karush, 1939).

3.4.1.3 $\rho(y, f(\mathbf{x}))$ – Margin–based loss function

By re-formulating the optimisation problem stated in (3.33) as

$$\begin{array}{ll} \min\limits_{\mathbf{w},w_0,\boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\mathbf{1}^T\boldsymbol{\xi} \\ s.t. & \xi_i \geq 1 - y_i\big(\mathbf{w}^T\phi(\mathbf{x}_i) + w_0\big) \quad i = 1,\ldots,N \\ & \boldsymbol{\xi} := \{\xi_i : \xi_i \geq 0,\, i = 1,\ldots,N\} \end{array} \tag{3.35}$$

it is able to integrate the linear constraints into the objective function and yields

$$\begin{array}{ll} \min\limits_{\mathbf{w},w_0} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum\limits_{i=1}^{N}\big(1 - y_i(\mathbf{w}^T\phi(\mathbf{x}) + w_0)\big)_+ \\ \text{with} & z_+ = \max(0, z) \end{array} \tag{3.36}$$

Clearly, problem (3.36) is equivalent to minimising the regularised risk function (3.13), with a *margin–based loss*

$$\rho(y, f(\mathbf{x})) := \max(0, 1 - yf(\mathbf{x})) \tag{3.37}$$

and the regulariser

$$\Lambda[f] := \mathbf{w}^T\mathbf{w}.$$

This margin–based loss is known as *soft–margin loss* (Bennett and Mangasarian, 1992; Cortes and Vapnik, 1995) and is commonly applied in maximum margin learning algorithms.

3.4.1.4 Implementations of the SVM optimisations

Although the SVM problem is convex and hence a unique optimal solution exists, there are always obstacles to obtain the solution. This section comments on some learning algorithms that tackled these obstacles.

Traditional learning methods, such as *interior point method* (Karmarkar, 1984) in convex optimisation exist for solving QP (e.g. SVM) problems. In general, such solutions in d variables cost a computational complexity that is $O(d^3)$. Hence, these traditional methods quickly become intractable when the dimension of feature space increases. In going to the dual formulation, the primal optimisation problem that involves minimising (3.33) over d variables, is turned into the dual representation (3.34) that has N variables. In addition, it allows the use of kernels and the typical solution can be applied efficiently to a feature space whose dimensionality exceeds the number of data samples, including the infinite feature space. However, the dual representation requires an amount of memory for storing kernels that is $O(N^2)$. This memory requirement limits the application of traditional QP techniques, especially to a large scale data (e.g. $N > 10^5$).

To reduce the memory required, Boser et al. (1992) proposed a method of solving the QP problem, which has since known as "*chunking*". The algorithm makes use of the fact that the value of the quadratic form is the same if you remove the rows and columns of the kernel matrix that corresponds to zero Lagrange multipliers. Therefore, the large QP problem can be broken down into a series of smaller QP problems, whose ultimate goal is to identify all of the non-zero Lagrange multipliers and discard all zero ones. The implementation is stated as follows. At every step the chunking algorithm solves a small QP problem that consists of the following examples: every non-zero Lagrange multiplier from the last step, and the m worst examples that violate the KKT conditions (if there are fewer than m examples that violate the constraints, all of the violating examples are added in). At the last step, no examples violate the KKT conditions and thus it solves the original QP problem. Although the chunking algorithm greatly reduces the size of the matrix, it cannot handle a large scale data, since even this reduced matrix cannot fit into memory.

Following this idea, Osuna et al. (1997) proved a theorem that suggests a whole new set of QP algorithms for SVMs. The theorem shows that by keeping a constant size matrix for every QP sub–problem, which implies at every step deleting m examples satisfying the KKT conditions and adding the same number of examples that violate the KKT conditions most, then the approach is guaranteed to converge. Platt (1999)

made use of this theorem and proposed the *sequential minimal optimisation* (SMO). The principle is to solve the smallest possible SVM problems that involves only two Lagrange multipliers at every step and iterates until convergence. The advantage of this approach lies in the fact that solutions for two Lagrange multipliers can be done analytically so that the numerical QP optimisation is avoided entirely. Meanwhile, it requires no extra matrix storage, which allows the training on arbitrarily sized data sets. As to time complexity, the experiment results in (Platt, 1999) demonstrated that SMO scaled somewhere between $O(N)$ and $O(N^2)$ while a traditional interior point method scaled somewhere between $O(N)$ and $O(N^3)$.

Returning to the primal optimisation problem of SVM, Shalev-Shwartz et al. (2007) proposed an approximate SVM solver (Pegasos) that is substantially faster than other existing SVM solvers. The method uses stochastic gradient descent on a small subset of data at each iteration, and bounds the norm of the weight vector within a pre-defined ball. As a stochastic gradient descent search, the solution is much faster than SMO proposed in (Platt, 1999). However, it is only an ϵ-accurate solution of SVM[4], in which the number of iterations required for achieving the solution is $O(\frac{R^2CN}{\epsilon})$ with $\|\mathbf{x}\| \leq R$. Therefore, its significant speed improvement is at the expense of loss of performance.

The readers are referred to (Schölkopf et al., 1999; Shalev-Shwartz et al., 2007) for more details of these solutions. Throughout the thesis, the SVM implementation is selected between *SVM–light* (Joachims, 1999) and *SVM–Multiclass* (Tsochantaridis et al., 2004).

3.4.2 Extending SVM to multi-class classification

The SVM is fundamentally a binary classifier. However, in practice we often have problems involving multiple classes. Various approaches have therefore been proposed for combining multiple binary–class SVMs in order to build a multi–class classifier. Note that some of these approaches are general, they can be applied to other binary classifiers (e.g. linear discriminant analysis) as well.

One commonly used method is "*one–versus–all*" proposed by (Duda and Hart, 1973). It involves constructing M classifiers, each of which solves a binary–class problem of separating examples in a particular class $\mathcal{C}_m$ from others. A new sample $\mathbf{x}$ is then assigned to class $\bar{\mathcal{C}}$ that satisfies

$$\bar{\mathcal{C}} := \arg\max_m \mathbf{w}_m^T \phi(\mathbf{x}) \tag{3.38}$$

This simple strategy however has two major weaknesses. One is the time complexity. In general for M separate SVMs, the computational complexity is somewhere between

[4]Let $J(\mathbf{w})$ denote the risk function given by equation (3.36), then an ϵ-accurate solution of $\mathbf{w}$ satisfies $J(\hat{\mathbf{w}}) < \min_{\mathbf{w}} J(\mathbf{w}) + \epsilon$.

$O(MN + N^2d)$ and $O(MN^2 + N^2d)$ (Bishop, 2006). If the number of output classes is considerably large, the runtime could be impractical to handle. For example, in machine translation each source sentence $\mathbf{f}$ and target sentence $\mathbf{e}$ is embedded into a linguistic feature space that has very high dimensions (usually $d \geq 50,000$) and the goal is to predict the target feature vector given the source. It can be regarded as a multi-class problem that contains a large number of output classes ($M \geq 50,000$), in which case "one–versus–all" is infeasible to apply. The other weakness is that the training sets maybe imbalanced. For instance, if we have 10 classes with equal numbers of training samples, then the individual classifiers are trained on 90% negative examples and only 10% positive ones, where the symmetry of the original problem is lost.

Another well–studied approach is "*one-versus-one*", that trains $M(M-1)/2$ different binary–class SVMs on all possible pairs of classes and classify the test examples according to which class has the highest number of "votes". However, the quadratic increase of time complexity with respective to M limits its application to large-scale multi-class problems.

"Smarter" variants, such as *directed acyclic graph* approach proposed by Platt et al. (2000), are considered as compromised strategies; and several experiments (Duan and Keerthi, 2005; Hsu and Lin, 2002) have been carried out to compare these strategies. Unfortunately, for different experiments different multi-class formations had the advantage of others; and finding an optimal solution for the application of SVMs to multi–class classification remains an open issue. In general, "one–versus–all" is widely used, in spite of its weaknesses and its practical limitations.

3.4.2.1 An extension of SVM – maximum margin regression (MMR)

Szedmak et al. (2006) introduced a novel methodology to multi-class classification, *maximum margin regression* (MMR), which is an extension of SVM and is a good candidate for large-scale multi-class problems. The principle is to project the output into a multi-dimensional subspace (vector) φ and find a maximum margin hyperplane based on the SVM technique. Take a three-class classification for example, the class labels can be projected into an indicator feature space, where the corresponding output vectors are formulated as $\varphi(y) = \{(0,0,1),(0,1,0),(1,0,0)\}$.

The learning procedure of MMR is illustrated in Figure 3.4. Mathematically, given a set of training examples $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, MMR learns a linear mapping $\mathbf{W} \in \mathbb{R}^{dim(\varphi)\times dim(\phi)}$ from the input feature space $\mathcal{H}_\phi$ to the output feature space $\mathcal{H}_\varphi$, by

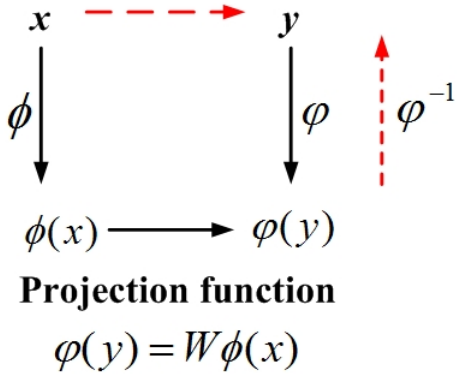


FIGURE 3.4: The illustration of the MMR prediction. Both the input and the output are projected into certain feature spaces where a linear operator $\mathbf{W}$ are then learnt from input to output.

solving the following optimisation problem

$$\begin{array}{ll} \min & \frac{1}{2}\mathbf{tr}(\mathbf{W}^T\mathbf{W}) + C\mathbf{1}^T\boldsymbol{\xi} \\ w.r.t & \{\mathbf{W}|\mathbf{W} : \mathcal{H}_\phi \rightarrow \mathcal{H}_\varphi, \mathbf{W} \text{ is a linear operator}\} \\ & \{\mathbf{b}|\ \mathbf{b} \in \mathcal{H}_\varphi,\ \mathbf{b} \text{ is a bias vector}\} \\ & \{\boldsymbol{\xi}|\boldsymbol{\xi} \in \mathbb{R}^N,\ \boldsymbol{\xi} \text{ is a slack or error vector}\} \\ s.t. & \langle\varphi(y_i), \mathbf{W}\phi(\mathbf{x}_i) + \mathbf{b}\rangle_{\mathcal{H}_\varphi} \geq 1 - \xi_i,\ i = 1, \ldots, N \\ & \boldsymbol{\xi} \geq \mathbf{0}. \end{array} \tag{3.39}$$

where $\mathbf{0}$ and $\mathbf{1}$ denote the vectors with components 0 and 1 respectively. This optimisation is an extension of (3.33) where the output labels are changed to vectors so as to encode the additional classes.

For a high dimensional feature space, the number of parameters in $\mathbf{W}$ is considerably large. A natural choice of reducing the number of weight parameters is to introduce dual variables $\boldsymbol{\alpha} = \{\alpha_i | i = 1, \ldots, N\}$ to the margin constraints and express the dual form of $\mathbf{W}$ in the light of Karush–Kuhn–Tucker conditions

$$\mathbf{W} = \sum_{i=1}^{N} \alpha_i \varphi(y_i)\phi(\mathbf{x}_i)^T \tag{3.40}$$

$\boldsymbol{\alpha}$ can then be induced by solving the dual optimisation problem

$$\begin{array}{ll} \min & \sum\limits_{i,j=1}^{N} \alpha_i\alpha_j\langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle\langle\varphi(y_i), \varphi(y_j)\rangle - \sum\limits_{i=1}^{N}\alpha_i \\ w.r.t & \{\alpha_i|\alpha_i \in \mathbb{R}\}_{i=1}^{N} \\ s.t & \sum\limits_{i=1}^{N} \alpha_i(\varphi(y_i))_t = 0,\ t = 1, \ldots, dim(\mathcal{H}_\varphi) \\ & 0 \leq \alpha_i \leq C, i = 1, \ldots, N \end{array} \tag{3.41}$$

where the inner products $\langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle$ and $\langle\varphi(y_i), \varphi(y_j)\rangle$ are usually defined implicitly via the kernel functions $K^{\phi}(\mathbf{x}_i, \mathbf{x}_j)$ and $K^{\varphi}(y_i, y_j)$.

Given the solution $\boldsymbol{\alpha}$, the feature prediction of a new example $\mathbf{x}$ is of the form

$$\varphi(y) = \sum_{i=1}^{N} \alpha_i \varphi(y_i) K^{\phi}(\mathbf{x}, \mathbf{x}_i) \tag{3.42}$$

and the pre-image of the example should be $\varphi^{-1}(\varphi(y))$.

In general the direct inversion is infeasible. But it is possible to obtain an approximation by exhaustively searching the output space such that

$$\begin{aligned} y &= \arg\max_{y' \in \mathbf{y}} \langle \varphi(y'), \mathbf{W}\phi(\mathbf{x})\rangle_{\mathcal{H}_{\varphi}} \\ &= \arg\max_{y' \in \mathbf{y}} \sum_{i=1}^{N} \alpha_i K^{\varphi}(y', y_i) K^{\phi}(\mathbf{x}, \mathbf{x}_i) \end{aligned} \tag{3.43}$$

It has been shown in (Szedmak et al., 2006) that this method is able to learn a structure (vector) output with the complexity of a binary SVM, which is independent of the size of the output. The advantage of its time complexity makes it suitable for large-scale multi-class problems or, the structured learning problems discussed in the next subsection.

3.4.3 Beyond multi-class – structured learning

In recent years, a newly emerging and excitingly attractive paradigm is solving problems involving complex outputs such as multiple dependent output variables and structured output spaces (e.g. ranking problems and label sequence learning). The term *structured learning* mentioned is different from that in the Bayesian networks (Cheng et al., 2002) and the multi-task learning models (Argyriou et al., 2008), where a latent structure in the *input domain* is assumed and learnt in parallel with the weight parameters $\mathbf{w}$. On the contrary, we interpret the term as "with the assumption of a latent structure in the *output domain*, applying an appropriate method for modelling the given data, which at the same time respect the output structure implicitly".

The term is vague, because the interpretation of "respect" can be broad, ranging from the method selection to the explicit design (or learning) of an output structure. For example, for text classification (document categorisation) problems, where a hierarchical structure of document categories (see Figure 3.5) exists although unknown, we can approximate it as a multi-class problem if we do not respect this fact. As an alternative, we can use the multi-category tree (created with prior knowledge) as a pre-defined output structure and use a structured prediction method to solve the problem.

Although there is no theoretic proof showing that the structured predictions are better than non-structured ones, the structured learning approaches have been applied to

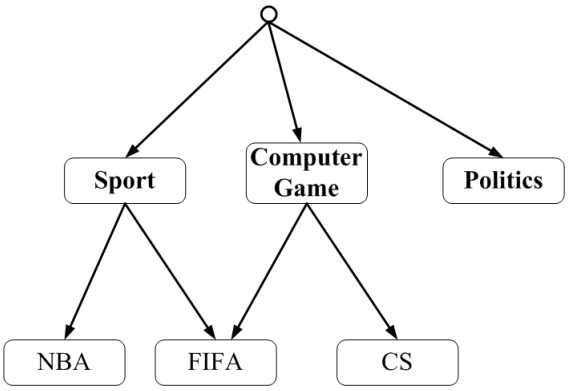


FIGURE 3.5: An example of the multi-category structure of the documents in text classification.

a comprehensive tasks, such as text classification (Rousu et al., 2005), part-of-speech tagging (Collins, 2002; Tsochantaridis et al., 2004; Wang and Shawe-Taylor, 2009) and machine translation (Wang and Shawe-Taylor, 2009); and the experimental results in (Tsochantaridis et al., 2004) and (Rousu et al., 2005) were better than those produced by non-structured predictions. Furthermore, in certain cases the traditional multi-class methods are difficult to apply, for example, the machine translation problems. As mentioned in Section 3.4.2, if we regard each target feature (target phrase) as a class, the traditional multi-class methods such as multi-class SVM will quickly become infeasible because of the rapid growth of output classes. On the contrary, the structured learning formulation such as that used in (Wang and Shawe-Taylor, 2009) yields tractable solutions.

Since most of the work in this thesis is inspired by the idea of structured learning. Below we introduce its general framework as well as several max-margin structured learning methodologies.

3.4.3.1 Structured learning framework

Suppose there is a set of training examples $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{r}_i)\}_{i=1}^{N}$, where $\mathbf{r}_i = \{r_{i1}, \ldots, r_{im}\} \in \mathbf{R}$ is a structured output such as sequence (of length m), strings (of length m), graphs (with m nodes) and so forth. The structured learning approach pursued is to learn a discriminant function

$$f(\mathbf{x}, \mathbf{r}; \mathbf{w}) := \mathbf{w}^T \phi(\mathbf{x}, \mathbf{r})$$

such that the correct output sequences will be assigned the highest scores

$$\mathbf{r}_i = \arg\max_{\bar{\mathbf{r}} \in \mathbf{R}} f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \quad \forall i.$$

This framework is known as *max-margin formulation* (Taskar et al., 2003) and is now commonly used in structured learning methods. It is equivalent to imposing the following set of non-linear constraints

$$\max_{\bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i} f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) < f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) \quad \forall i \tag{3.44}$$

which can be converted to $N(|\mathbf{R}| - 1)$ linear constraints

$$f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) < f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) \quad \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \tag{3.45}$$

3.4.3.2 Structured SVM

By adding the set of linear constraints (3.45) to an SVM, Tsochantaridis et al. (2004) generalised the multi-class SVMs to the broader problems of learning structured responses and proposed the *structured SVM* approach. This approach generalises the maximum margin principle employed in SVMs to the more general case

$$\begin{array}{ll} \min\limits_{\mathbf{w}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ s.t. & f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) - f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \geq 1 \\ & \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \end{array} \tag{3.46}$$

with the intension of separating each example $(\mathbf{x}_i, \mathbf{r}_i)$ from $|\mathbf{R}| - 1$ "pseudo-examples" $(\mathbf{x}_i, \bar{\mathbf{r}})$ with margin 1.

Similarly, to allow errors in the training set, the method introduces slack variables $\boldsymbol{\xi}$ and suggests to optimise a soft-margin criterion

$$\begin{array}{ll} \min\limits_{\mathbf{w}, \boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\mathbf{1}^T\boldsymbol{\xi} \\ s.t. & f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) - f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \geq 1 - \xi_i \\ & \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \\ & \boldsymbol{\xi} := \{\xi_i | \xi_i \geq 0,\ i = 1, \ldots, N\} \end{array} \tag{3.47}$$

So far, the model implicitly considers the zero-one classification loss. This is inappropriate for problems like text classification where the output domain indeed has a latent structure. Roughly speaking, some labels (classes) in the output domain are closer and these relations should be revealed in the optimisation. In this sense, the structured SVM designs a matrix $\Delta(\mathbf{r}_i, \bar{\mathbf{r}})$ to evaluate the "distance" between a pseudo label $\bar{\mathbf{r}}$ and the true one $\mathbf{r}_i$ and the problem is finalised as the following optimisation problem

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi}} \quad & \tfrac{1}{2}\mathbf{w}^T\mathbf{w} + C\mathbf{1}^T\boldsymbol{\xi} \\ s.t. \quad & f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) - f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \geq \Delta(\mathbf{r}_i, \bar{\mathbf{r}}) - \xi_i \\ & \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \\ & \boldsymbol{\xi} := \{\xi_i | \xi_i \geq 0,\, i = 1, \ldots, N\} \end{aligned} \tag{3.48}$$

Theoretically, violating a margin constraint involving two similar labels $\bar{\mathbf{r}} \neq \mathbf{r}_i$ can be penalised less (represented by a smaller margin $\Delta(\mathbf{r}_i, \bar{\mathbf{r}})$) where in contrast it should be penalised more severely. This idea is referred to as "re-scaling the margin". An alternative approach, namely "re-scaling the slack variables", is also presented in (Tsochantaridis et al., 2004) that changes the constraints in (3.48) to $f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) - f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \geq 1 - \frac{\xi_i}{\Delta(\mathbf{r}_i, \bar{\mathbf{r}})}$.

The structured SVM addresses the complementary issue of problems involving structured and interdependent outputs. Meanwhile, its empirical results on the natural language parsing tasks verified that the algorithm is tractable (Tsochantaridis et al., 2004). However, the model has a total of $N(|\mathbf{R}| - 1)$ constraints, where $|\mathbf{R}|$ may grow exponentially in the length of $\mathbf{r}$. Although Tsochantaridis et al. (2004) proved that the optimisation problem can be solved with a runtime polynomial in N which is independent of $|\mathbf{R}|$, the lengthy training time still restricts its application in some large scale problems like machine translation.

3.4.3.3 Structured prediction with extra-gradient method

Instead of using the $N(|\mathbf{R}| - 1)$ linear constraints (3.45), Taskar et al. (2006) proposed another structured prediction model, which integrates the non-linear constraints (3.44) into the risk function and adds a standard L_2 weight penalty to avoid over-fitting. The resulting optimisation problem is of the form

$$\min_{\mathbf{w}} \quad \tfrac{1}{2C}\|\mathbf{w}\|^2 + \sum_i \Big\{ \max_{\bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i} \big(f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) + \Delta(\mathbf{r}_i, \bar{\mathbf{r}}) \big) - f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) \Big\} \tag{3.49}$$

The spirit of this method is to introduce a set of $N|\mathbf{R}|$ auxiliary parameters $\mathbf{Z} = \{\mathbf{z}_i : \|\mathbf{z}\| \leq 1\ \mathbf{z}_i \in \mathbb{R}^{|\mathbf{R}|}\}_{i=1}^N$ to reformulate the $\max_{\bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i}$ operation in (3.49) as a *linear programming* problem

$$\begin{aligned} \max_{\mathbf{Z}} \quad & \sum_i \Big\{ \sum_{j=1, j \neq i}^{|\mathbf{R}|} z_{ij} \big(f(\mathbf{x}_i, \bar{\mathbf{r}}_j; \mathbf{w}) + \Delta(\bar{\mathbf{r}}_j, \mathbf{r}_i) \big) - f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) \Big\} \\ s.t. \quad & \textstyle\sum_j z_{ij} \leq 1 \quad \forall i. \end{aligned} \tag{3.50}$$

which then replaces $\max_{\bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i}$ in (3.49) and yields a joint convex optimisation problem – a saddle point optimisation

$$\begin{array}{ll} \min_{\mathbf{w}} \max_{\mathbf{Z}} & \frac{1}{2C}\|\mathbf{w}\|^2 + \sum_i \Big\{ \sum_{j=1, j\neq i}^{|\mathbf{R}|} z_{ij}\big(f(\mathbf{x}_i, \bar{\mathbf{r}}_j; \mathbf{w}) + \Delta(\bar{\mathbf{r}}_j, \mathbf{r}_i)\big) - f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w})\Big\} \\ s.t. & \sum_j z_{ij} \leq 1 \quad \forall i. \end{array} \tag{3.51}$$

To obtain the solution, the authors suggested using the extra-gradient strategy (Korpelevich, 1976), which will be introduced in Section 3.5.

Compared with the learning algorithm of structured SVM, whose runtime is polynomial in N, the algorithm used in (Taskar et al., 2006) (namely *Exponentiated Gradient*, which is a derivative of the extra-gradient method) has sublinear convergence rate guarantees. In the case where the output structure is not too complicated (represented by a smaller $|\mathbf{R}|$), this structured prediction model can be faster than a structured SVM.

3.4.3.4 Structured classification via linear programming

An alternative max-margin structured prediction method is proposed by Wang and Shawe-Taylor (2009), who replaced the L_2 weight penalty in (3.47) with an L_1 regularisation so as to make it a linear programming problem. To simplify the solution, they also constrained $\mathbf{w}$ to be non-negative and the optimisation problem was given by the formula

$$\begin{array}{ll} \min_{\mathbf{w}, \boldsymbol{\xi}} & \|\mathbf{w}\|_1 + C\mathbf{1}^T\boldsymbol{\xi} \\ s.t. & f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) - f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) \geq 1 - \xi_i \\ & \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \\ & \boldsymbol{\xi} := \{\xi_i | \xi_i \geq 0,\ i = 1, \ldots, N\} \end{array} \tag{3.52}$$

To facilitate the training speed, the authors proposed using *column generation* with the extra-gradient strategy to solve the optimisation problem. However, no direct comparison with other structured prediction methods were given in their experiments. For memory usage, the authors argued that the proposed method scaled better than the previous QP-based models, but no direct evidence is provided in (Wang and Shawe-Taylor, 2009).

3.4.3.5 Kernel–based structured prediction — H-M^3

The weakness of formation (3.48) is that it suffers from the possible high-dimensionality of the feature vector. By turning it into the dual space with variables $\boldsymbol{\alpha} = \{\alpha_{ij} \geq 0,\ i =$

$1, \ldots N;\ j = 1, \ldots, |\mathbf{R}|\}$, the dual representation is expressed as

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{N} \sum_{j=1}^{|\mathbf{R}|} \alpha_{ij} \Delta(\mathbf{r}_i, \bar{\mathbf{r}}_j) - \tfrac{1}{2} \sum_{i,j} \sum_{i',j'} \alpha_{ij} \alpha_{i'j'} K(i,j,i',j') \\ s.t. \quad & \sum_{j=1}^{|\mathbf{R}|} \alpha_{ij} \leq C,\ \forall i, \end{aligned} \tag{3.53}$$

where kernel $K(i, j, i', j')$ is defined by

$$K(i,j,i',j') = \Delta\phi(\mathbf{x}_i, \mathbf{r}_i, \bar{\mathbf{r}}_j)^T \Delta\phi(\mathbf{x}_{i'}, \mathbf{r}_{i'}, \bar{\mathbf{r}}_{j'}) \tag{3.54}$$

with $\Delta\phi(\mathbf{x}_i, \mathbf{r}_i, \bar{\mathbf{r}}_j) = \phi(\mathbf{x}_i, \mathbf{r}_i) - \phi(\mathbf{x}_i, \bar{\mathbf{r}}_j)$.

In (3.53) there might be exponentially many dual variables (if $|\mathbf{R}|$ grows exponentially in the length of $\mathbf{r}$), one for each pseudo-example. Although (Tsochantaridis et al., 2004) has provided an approximate solution with a polynomial number of support vectors, the training time is still expensive for large-scale learning problems.

To further reduce the time complexity, Rousu et al. (2005) proposed a variant of *hierarchical maximum margin markov network* (H-M^3), that involves only a polynomial-sized optimisation problem. The basic assumption of this method is that the distance matrix is example-wise decomposable. That is, the distance measure between $\mathbf{r}$ and $\bar{\mathbf{r}}$ is computed in an additive manner. One example is the Hamming distance $\Delta(\mathbf{r}, \bar{\mathbf{r}}) = \sum_{k=1}^{m} \delta(r_k \neq \bar{r}_k)$ where δ represents an indicator function. With this assumption a graph $\mathbf{E}$ is constructed for the output domain with maximum m^2 edges $e = (k, k')$ and the output $\mathbf{r}$ can then be decomposed into a set of edges with $\mathbf{r}_e = (r_k, r_{k'}) \in \mathcal{R}_e$. Suppose each node k has $\mathcal{N}_k$ edges, the distance between example $\mathbf{r}$ and $\bar{\mathbf{r}}$ on edge e is then computed by

$$\Delta_e(\mathbf{r}, \bar{\mathbf{r}}) = \frac{1}{\mathcal{N}_k} \delta(r_k \neq \bar{r}_k) + \frac{1}{\mathcal{N}_{k'}} \delta(r_{k'} \neq \bar{r}_{k'}) \quad \forall e = (k, k') \tag{3.55}$$

and the total distance between $\mathbf{r}$ and $\bar{\mathbf{r}}$ can be obtained by summing over all *edge-distances*. Similarly, the distance of example $\mathbf{r}$ on edge e is of the form

$$\Delta_e(\mathbf{r}_e) = \sum_{\bar{\mathbf{r}}_e \in \mathcal{R}_e} \big(\frac{1}{\mathcal{N}_k} \delta(r_k \neq \bar{r}_k) + \frac{1}{\mathcal{N}_{k'}} \delta(r_{k'} \neq \bar{r}_{k'}) \big) \quad \forall e = (k, k') \tag{3.56}$$

Making use of this output graph, the core of H-M^3 is to replace the original $|\mathbf{R}|$ dual variables $\boldsymbol{\alpha}$ with a reduced set $\boldsymbol{\mu} = \{\mu_e | e \in \mathbf{E}\}$ that was named *edge-marginals* in (Taskar et al., 2003)

$$\mu_e(i, \mathbf{r}_e) = \mu_e(i, r_k, r_{k'}) = \sum_{\{j | \bar{\mathbf{r}}_{je} = \mathbf{r}_e\}} \alpha_{ij} \tag{3.57}$$

where each e-edge-marginal of example i is the sum of $\{\alpha_{ij}\}$ for those pseudo-examples $\{\bar{\mathbf{r}}_j\}$ that have the same labels on this edge. Meanwhile, the joint feature vector $\phi(\mathbf{x}_i, \mathbf{r}_i)$

is decomposed via the *orthogonal feature representation*[5]

$$\phi(\mathbf{x}_i, \mathbf{r}_i) = (\phi_e(\mathbf{x}_i, \mathbf{r}_{ie}))_{e \in \mathbf{E}}, \tag{3.58}$$

so that there is a block for each edge, which in turn, is divided into blocks for a specific edge-labeling pairs $(e, \mathbf{r}_e)$

$$\phi_e(\mathbf{x}_i, \mathbf{r}_{ie}) = (\phi_e^{\mathbf{r}_e}(\mathbf{x}_i, \mathbf{r}_{ie}))_{\mathbf{r}_e \in \mathcal{R}_e} \tag{3.59}$$

with $\phi_e^{\mathbf{r}_e}(\mathbf{x}_i, \mathbf{r}_{ie}) = \delta(\mathbf{r}_e = \mathbf{r}_{ie})\phi(\mathbf{x}_i)$.

Utilising the above feature map, the kernel expression (3.54) is also decomposed as a combination of *edge-kernels*

$$\begin{aligned} K(i,j,i',j') &= \sum_{e \in \mathbf{E}} \Delta\phi_e(\mathbf{x}_i, \mathbf{r}_{ie}, \bar{\mathbf{r}}_{je})^T \Delta\phi_e(\mathbf{x}_{i'}, \mathbf{r}_{i'e}, \bar{\mathbf{r}}_{j'e}) \\ &= \sum_{e \in \mathbf{E}} [\phi_e(\mathbf{x}_i, \mathbf{r}_{ie}) - \phi_e(\mathbf{x}_i, \bar{\mathbf{r}}_{je})]^T [\phi_e(\mathbf{x}_{i'}, \mathbf{r}_{i'e}) - \phi_e(\mathbf{x}_{i'}, \bar{\mathbf{r}}_{j'e})] \\ &= \sum_{e \in \mathbf{E}} K(i, \mathbf{r}_e, i', \mathbf{r}'_e) \end{aligned} \tag{3.60}$$

In other words, the feature vector is taken into account if and only if the pseudo-example and the correct one have different labels on edge e.

Finally in order to ensure that the edge-marginals correspond to a valid $\boldsymbol{\alpha}$, additional constraints are required (Taskar et al., 2003). That is, if two edges share a node p, they need to have equal node-margins. Mathematically, for all sharing-node edge pairs $\mathbf{E}_2 = \{(e, e') \in \mathbf{E} \times \mathbf{E} | e = (k, p), e' = (p, k')\}$, the edge-marginals should satisfy

$$\sum_k \mu_e(i, r_k, r_p) = \sum_{k'} \mu_{e'}(i, r_p, r_{k'}) \ \forall (e, e') \in \mathbf{E}_2. \tag{3.61}$$

By utilising expression (3.56) to (3.61), the optimisation problem (3.53) is reformulated as

$$\begin{aligned} \max_{\boldsymbol{\mu} > 0} \quad & \sum_{e \in \mathbf{E}} \sum_{i=1}^{N} \mu_e(i, \mathbf{r}_{ie}) \Delta_e(\mathbf{r}_{ie}) - \tfrac{1}{2} \sum_{e \in \mathbf{E}} \sum_{i,i'=1}^{N} \sum_{\mathbf{r}_e, \mathbf{r}'_e} \mu_e(i, \mathbf{r}_e) K_e(i, \mathbf{r}_e, i', \mathbf{r}'_e) \mu_e(i, \mathbf{r}'_e) \\ s.t. \quad & \sum_{\mathbf{r}_{ie}} \mu(i, \mathbf{r}_{ie}) \leq C \ \forall i, e \in \mathbf{E} \\ & \sum_k \mu_e(i, r_k, r_p) = \sum_{k'} \mu_{e'}(i, r_p, r_{k'}) \ \forall (e, e') \in \mathbf{E}_2. \end{aligned} \tag{3.62}$$

which is now a polynomial-sized optimisation problem. The solution of (3.62) can then be obtained by *conditional subspace gradient ascent*, as suggested in (Rousu et al., 2005).

Although H-M^3 has a very nice framework in modelling the output structure, the time complexity is troublesome. The experiments carried out in (Rousu et al., 2005) were two

[5]The readers are referred to (Rousu et al., 2006) for the detailed description of this feature representation.

text classification tasks that involved 34 output labels and only up to 2500 examples. However, the proposed method took almost six hours to get the optimisation solution. In this case, it is difficult to apply to large scale problems like machine translation, which usually involves in processing hundreds of thousands of examples.

3.4.3.6 Perceptron-based structured learning

Input of the learner: The samples $\mathcal{S} = \{(\mathbf{x}_i, \mathbf{r}_i)\}_{i=1}^{N}$, learning rate η
Initialisation: $t = 0$; $\mathbf{w}^t = \mathbf{0}$;
Repeat
  for $i = 1, 2, \ldots, N$ **do**
    read input: $\phi(\mathbf{x}_i, \mathbf{r}_i) \in \mathbb{R}^{d \cdot |\mathbf{R}|}$;
    $V = \max_{\bar{\mathbf{r}}} \langle \mathbf{w}^t, \phi(\mathbf{x}_i, \bar{\mathbf{r}}) \rangle$
    $\bar{\mathbf{r}}^* = \arg\max_{\bar{\mathbf{r}}} \langle \mathbf{w}^t, \phi(\mathbf{x}_i, \bar{\mathbf{r}}) \rangle$
    if $\langle \mathbf{w}^t, \phi(\mathbf{x}_i, \mathbf{r}_i) \rangle < V$ **then**
      $\mathbf{w}^{t+1} = \mathbf{w}^t + \eta\big(\phi(\mathbf{x}_i, \mathbf{r}_i) - \phi(\mathbf{x}_i, \bar{\mathbf{r}}^*)\big)$
      $t = t + 1$
    end if
  end for
until converge
Output of the learner: $\mathbf{w}^{t+1} \in \mathbb{R}^{d \cdot |\mathbf{R}|}$

TABLE 3.2: Pseudo-code of the perceptron–based learning algorithm.

Apart from the above structured learning methodologies, Collins (2002) proposed a discriminative training method for sequence labeling, which is also related to structured prediction. Although not explicitly pointed out in (Collins, 2002), the proposed method aims at solving the following optimisation problem

$$\min_{\mathbf{w}} \sum_{i} \Big\{ \max_{\bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i} f(\mathbf{x}_i, \bar{\mathbf{r}}; \mathbf{w}) - f(\mathbf{x}_i, \mathbf{r}_i; \mathbf{w}) \Big\} \tag{3.63}$$

It is easy to show that this is a special case of (3.49), which ignores the regularised term and the output structure (i.e. $\Delta(\mathbf{r}_i, \bar{\mathbf{r}}) = 1 \; \forall \bar{\mathbf{r}}$). In this case, the optimisation solution can be derived using a perceptron-based learning algorithm, which is depicted in Table 3.2. The advantage of this method is its time complexity, which is linear in N. Although finding $\bar{\mathbf{r}}^*$ requires searching the whole output space that might have exponentially many classes, in sequence labeling this exhaustive search can be avoided by approximate solutions such as Viterbi decoding. Therefore, its training procedure is substantially faster than the above approaches. This time efficiency is also confirmed by the part-of-speech tagging experiments carried out in (Wang and Shawe-Taylor, 2009).

Despite the great advantage of its computational complexity, this method tends to over-fit the training data since it lacks a regularisation term. Moreover, it does not consider

any relationship in the output domain, which might cause a potential loss in classification accuracy and tunability.

3.4.4 "Respect" the output structure – the distance measurement

Theoretically, all the above approaches allow arbitrary structures in the output domain. Nevertheless, allowing any output structure does not mean respecting it. Take the method in (Wang and Shawe-Taylor, 2009) for example, the model in (3.52) can be reformulated as

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi}} \quad & \|\mathbf{w}\|_1 + C\sum_{i=1}^{N}\xi_i \\ s.t. \quad & f(\mathbf{x}_i,\mathbf{r}_i;\mathbf{w}) - f(\mathbf{x}_i,\bar{\mathbf{r}};\mathbf{w}) \geq \Delta(\mathbf{r}_i,\bar{\mathbf{r}}) - \xi_i \\ & \forall i, \forall \bar{\mathbf{r}} \in \mathbf{R} \setminus \mathbf{r}_i \\ & \boldsymbol{\xi} := \{\xi_i | \xi_i \geq 0,\ i = 1,\ldots,N\} \end{aligned} \tag{3.64}$$

Because no relationships between a correct example $\mathbf{r}_i$ and pseudo-examples $\bar{\mathbf{r}}$ are considered, the model implicitly applies *zero-one loss* as the distance measure $\Delta(\mathbf{r}_i,\bar{\mathbf{r}}) = \delta(\mathbf{r}_i = \bar{\mathbf{r}})$. It gives loss of 1 if the complete output structure is not labeled correctly, even if only a single label was predicted incorrectly. In effect it is equivalent to "one-versus-all" strategy, if we view each unique output sequence as a class. In their part-of-speech tagging experiments, the results were worse than a conditional random field (CRF) model, which might due to the lack of respect on the output structure as mentioned.

Respecting the output structure is equivalent to designing an appropriate distance matrix. The *Hamming distance* used in (Taskar et al., 2006) has a nice property in measuring the distance between label sequences and is an obvious first choice as the distance measurement in structured prediction tasks. However, the output hierarchy is not reflected in any way in this loss. For example, using the Hamming distance to measure the multi-category structure in Figure 3.5 is equivalent to measuring a flat tree where label "Sport" is as important as label "NBA". In this way, the latent connects between labels are lost.

A good distance measurement for hierarchial structures should respect the labels as well as their connections. One measurement that has these properties was given in (Cesa-bianchi et al., 2004). It penalises the first mistake along a path from root to a node

$$\Delta(\mathbf{r}_i,\bar{\mathbf{r}}) = \sum_k c_k \delta\big(r_{ik} \neq \bar{r}_k \ \&\ r_{ih} = \bar{r}_h \ \forall h \in anc(k)\big)$$

where $anc(k)$ denotes the set of ancestors of node k and the coefficients $0 \leq c_k \leq 1$ are used for down-scaling the loss when going deeper in the subtree. Unfortunately, using this path measurement requires a clear output structure (i.e. a fixed output structure for

all examples), which is difficult to obtain in many cases such as part-of-speech tagging and machine translation.

Other distance measurements, such as uni-category hierarchical classification (Hofmann et al., 2003) and weighted hierarchical loss (Cai and Hofmann, 2004) are also suggested. But as shown in (Rousu et al., 2006), for different prediction tasks different distance measurements were superior to others. Therefore, which measurement suits which task is somewhere in the twilight zone.

Alternatively, is it possible to learn the distance matrix automatically, similar to what has been done in multi-task learning? This idea might result in a new structured learning framework, which could be of great interest to ML researchers.

3.4.4.1 Problems of structured learning

Since structured learning has to consider a complex structure in the output domain, the models are usually complicated and one of the most weaknesses being criticised is its runtime. Although the perceptron-based learning approach has a good time complexity, much need should be done in reducing the risk in over-fitting. Furthermore, how to respect the output structure is more art than science; designing a distance matrix for a specific task requires not only mathematical models but also a deep knowledge for the problem the task attempts to solve.

3.4.5 Summary

By formulating the function $\rho(y, f(\mathbf{x}))$ in different margin–based losses, the maximum margin classifier can spawn a set of discriminative models such as SVMs, max-margin markov networks (Taskar et al., 2003), maximum margin regression (Szedmak et al., 2006) and structured SVMs (Tsochantaridis et al., 2004). This margin–based classifier family has been one of the most important components in machine learning and the classifiers have been applied successfully to numerous areas, ranging from financial market (Tay and Cao, 2001) that detects the trends of stock data (*financial time series forecasting*) with SVMs, to brain activity prediction (Ni et al., 2008), which utilises max-margin techniques to predict human's activities based on their brain scans.

Nevertheless, the relatively large time complexity usually limits its application to large scale data sets and the compromised solutions are still under development. There is a gap between the time efficiency and the classification performance, which is expected to be filled up by the developers of machine learning.

3.5 Learning algorithms

Supervised learning problems always result in minimising a risk function (3.13) with variant losses, which build up different optimisation problems, such as *minimisation* problems (e.g. optimisation (3.9)) and *minimax* (*saddle point*) optimisation problems (Benzi et al., 2005). The requirements of solving these problems resort to *learning algorithms*, which are also different from one to another. For example in minimisation optimisation, if the global minimum exists, it is able to be found by direct solutions, like (3.10) and (3.12). However, the practice still limits the use of these solutions, such as the number of input variables, the convexity of the risk function and the number of constraints. In these cases, the problems can still be solved, alternatively by *iterative learning algorithms*. This section concerns three iterative learning algorithms that act as supplements of direct solutions and they are also the basis of the proposed work in this thesis. In addition, their convergence properties are also commented although not proved.

3.5.1 Iterative learning algorithms

In computational mathematics, an *iterative learning algorithm* attempts to solve a problem (e.g. system of equations) by finding successive approximations to the solution starting from an initial guess. Table 3.3 shows a general procedure of the iterative methods, in which the vector $\Delta(\mathbf{w})$ is named *search direction* and the positive parameter η denotes the *step length*. For simplicity, in Table 3.3 $\Delta(\mathbf{w})$ depends on the previous update $\mathbf{w}_t$ only, but in practice (e.g. discrete dynamic system) this rule is not necessary to hold.

Goal: minimise the risk function $J(\mathbf{w})$.
1) **Initialise** a starting value $\mathbf{w}_0$ for $\mathbf{w}$
2) **for** $t = 1 \cdots \tau$ **do**
a) Compute the new weight vector $\mathbf{w}_{t+1}$ using the update rule $\mathbf{w}_{t+1} = \mathbf{w}_t + \eta_t \Delta(\mathbf{w}_t)$, $\eta_t > 0$
b) Stop if reaching the stopping criterion
end for
Output: weight vector $\mathbf{w}_{t+1}$

TABLE 3.3: Pseudo-code of the iterative learning algorithm.

The purpose of optimisation is to find $J^* = \min_{\mathbf{w}} J(\mathbf{w})$, hence the goal of the learning algorithm is to make $J(\mathbf{w}_t) \to J^*$ as $t \to \infty$. This results in characterising both search direction and step length. Over the past decade, there exists a variety of candidate choices for search direction and step length, making iterative learning algorithms a big family. Naturally the step length is chosen as a constant, or the minimiser of $J(\mathbf{w} + \eta\Delta(\mathbf{w}))$

$$\bar{\eta} := \arg\min_{\eta} J(\mathbf{w} + \eta\Delta(\mathbf{w})),\ \eta > 0$$

which is known as *line search*. Therefore, this section concentrates on the decisions of $\Delta(\mathbf{w})$.

3.5.2 Gradient descent method

Gradient descent, also known as *steepest descent*, is a learning algorithm to find a local minimum of a function using its negative gradient.

To choose a direction $\Delta(\mathbf{w})$ it requires that

$$J(\mathbf{w}_{t+1}) < J(\mathbf{w}_t) \tag{3.65}$$

unless $\mathbf{w}_t$ is optimal.

If $J(\mathbf{w})$ is a differentiable convex function, that means, for all $\mathbf{w}, \mathbf{v} \in \mathbb{R}^d$,

$$J(\mathbf{w} + \mathbf{v}) \geq J(\mathbf{w}) + \nabla J(\mathbf{w}) \cdot \mathbf{v}$$

where $\bigtriangledown J(\mathbf{w})$ denotes the gradient of $J(\mathbf{w})$. Hence, to fulfill (3.65) the choice of $\Delta(\mathbf{w})$ should satisfy that

$$\Delta(\mathbf{w}) \nabla J(\mathbf{w}) < 0,$$

from which a natural choice is to take $\Delta(\mathbf{w})$ to be the negative gradient of $J(\mathbf{w})$

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla J(\mathbf{w}_t) \tag{3.66}$$

Update (3.66) is the essence of the gradient descent. It reflects our intuition that if we take a little step down the gradient, $J(\mathbf{w})$ will decrease. The convergence of this algorithm is provided in (Bertsekas, 1999), saying that if η_t is chosen via line search and $J(\mathbf{w})$ is strongly convex, then it converges to the global minimum of $J(\mathbf{w})$. Otherwise if $J(\mathbf{w})$ has more than one minimum (i.e. local minimums exist), the algorithm converges to a local minimum, depending on the initial value of $\mathbf{w}$.

The gradient descent algorithm is natural and simple, but has its weaknesses

1. If the curvature in different directions is very different, the algorithm could take many iterations to converge towards a local minimum.
2. Finding the optimal step length η_t can be time–consuming, while in contrast using a fixed η may yield poor results.
3. Since the gradient descent method always converges towards the most important direction of minimisation problems, it converges for saddle point problems only under a strong concavity–convexity assumption of the objective function.

To address these problems, more powerful algorithms are required, such as *Newton's method* and *conjugate gradient technique*. The readers are referred to (Bertsekas, 1999) for further details of these advanced learning approaches.

3.5.3 Sequential learning – online gradient method

Given the risk function of the form

$$J(f|S) = \sum_{i=1}^{N} \rho(y_i, f(\mathbf{x}_i)) + \frac{\lambda}{2}\|f\|^2$$

rather than computing the gradient of the batch of input samples, one can select one sample at a time. Suppose at time t the sample received is i_t, then using the stochastic approximation of $J(f|S)$

$$J_{i_t}(f|S) := \rho(y_{i_t}, f(\mathbf{x}_{i_t})) + \frac{\lambda}{2}\|f\|^2,$$

and setting the gradient

$$g_t := \partial_f J_{i_t}(f_t|S) = \partial_f \rho(y_{i_t}, f_t(\mathbf{x}_{i_t})) + \lambda f_t,$$

we obtain a simple rule of *online gradient* update

$$f_{t+1} = f_t - \eta_t g_t \tag{3.67}$$

with η_t denoting the step length at time t.

Updating rule (3.67) is similar to that of (3.66), while the gradient is computed only at one data point. The pseudo-code of this online gradient method is depicted in Table 3.4. This method is usually applied when the entire input is unavailable from the start or the computation of gradient for the batch learning algorithms is too expensive. A variety of online learning methods is born out of this simple formulation, such as ALMA in (Gentile, 2001), NORMA in (Kivinen et al., 2004) and modified Perceptron ALMA in (Cesa-Bianchi and Lugosi, 2006). Instead of using only one sample at a time, a more compromised approach is to draw a few samples at random at a time, which is referred to as the *stochastic gradient method*.

3.5.4 Extra–gradient method

The gradient descent method is one of the simplest and most natural methods for finding minimisation. However, the properties of the saddle point problems such as "non–stability" usually make it hard (or impossible) for the gradient methods to converge

Goal: minimise the risk function $J(\mathbf{w})$.
1) **Initialise** a starting value $f_0 = 0$
2) **for** $t = 1 \cdots \tau$ **do**
a) Draw a random sample $(\mathbf{x}_{i_t}, y_{i_t})$
b) Predict $f_t(\mathbf{x}_{i_t})$ and incur loss $\rho(y_{i_t}, f(\mathbf{x}_{i_t}))$
d) Update $f_{t+1} = f_t - \eta_t g_t$
d) Stop if reaching the stopping criterion
end for
Output: evaluation function f_{t+1}

TABLE 3.4: Pseudo-code of the online gradient update.

with any length of step (Korpelevich, 1976). A better way of solving this problem is to modify the gradient method itself by using extrapolation, yielding the *extra–gradient method* (Korpelevich, 1976).

A saddle point problem (i.e. min–max problem) is an optimisation problem of the form

$$\min_{\mathbf{w}} \max_{\mathbf{z} \in \mathcal{Z}} \mathcal{L}(\mathbf{w}, \mathbf{z}) \tag{3.68}$$

where any solution $(\mathbf{w}^*, \mathbf{z}^*)$ is a so called *saddle point*.

The extra–gradient solution of this optimisation problem consists of two very simple steps in an iteration

$$\text{(Prediction)} \begin{cases} \bar{\mathbf{w}}^{t+1} = \mathscr{P}_R(\mathbf{w}^t - \eta \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}^t, \mathbf{z}^t)) \\ \bar{\mathbf{z}}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{w}^t, \mathbf{z}^t)) \end{cases} \tag{3.69}$$

$$\text{(Correction)} \begin{cases} \mathbf{w}^{t+1} = \mathscr{P}_R(\mathbf{w}^t - \eta \nabla_{\mathbf{w}} \mathcal{L}(\bar{\mathbf{w}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \\ \mathbf{z}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\bar{\mathbf{w}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \end{cases} \tag{3.70}$$

where $\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{w}, \mathbf{z})$ represents the partial derivative of the cumulative loss $\mathcal{L}$ with respect to $\mathbf{x}$ and $\mathscr{P}(\mathbf{x})$ denotes the projection of vector $\mathbf{x}$.

The pseudo-code of the extra–gradient method is described in Table 3.5. Under mild conditions, this method is guaranteed to converge linearly to a solution of $\mathbf{w}^*$ and $\mathbf{z}^*$ (Korpelevich, 1976).

3.5.5 Matters of iterative learning algorithms

There are tremendous other iterative learning algorithms, which can take days to named. But all of them have two matters in common: *convergence rate* and *generative error*

Goal: minimise the cumulative loss function $\mathcal{L}(\mathbf{w}, \mathbf{z})$.

1) **Initialise** a starting value $\mathbf{w}_0$ for $\mathbf{w}$ and $\mathbf{z}_0$ for $\mathbf{z}$

2) **for** $t = 1 \cdots \tau$ **do**

a) **Prediction step** for $\mathbf{w}$ and $\mathbf{z}$ using the updating rule

$$\begin{cases} \bar{\mathbf{w}}^{t+1} = \mathscr{P}_R(\mathbf{w}^t - \eta \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}^t, \mathbf{z}^t)) \\ \bar{\mathbf{z}}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{w}^t, \mathbf{z}^t)) \end{cases}$$

b) **Correction step** for $\mathbf{w}$ and $\mathbf{z}$ using the updating rule

$$\begin{cases} \mathbf{w}^{t+1} = \mathscr{P}_R(\mathbf{w}^t - \eta \nabla_{\mathbf{w}} \mathcal{L}(\bar{\mathbf{w}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \\ \mathbf{z}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\bar{\mathbf{w}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \end{cases}$$

c) Stop if reaching the stopping criterion

until converge

Output: weight vectors $\mathbf{w}_{t+1}$ and $\mathbf{z}_{t+1}$

TABLE 3.5: Pseudo-code of the extra–gradient learning algorithm.

bound. A better generative error bound is always important to estimate the prediction of a new sample, while in practice it may be sacrifice for fast convergence, especially for large-scale data sets.

3.6 Machine learning for machine translation

Originated from computer science, machine learning is now one of the most fundamental research areas and has a long, successful history. On the contrary, machine translation is still in an exploratory stage, the traditional statistical methods applied are unable to mimic the complicated generative process of human translations. By exploring a further wide field of linguistic features, the applications of ML techniques have achieved significant developments in adequacy and fluency of the translations. Recent MT systems, especially SMT systems, employ a wide range of ML technologies, such as maximum entropy approaches (Berger et al., 1996; Koehn et al., 2005; Zens and Ney, 2006), regression models (Cortes et al., 2005; Wang et al., 2007), maximum margin–based techniques (Giménez and Màrquez, 2007; Wang and Shawe-Taylor, 2009) and so fourth. The applications turn out to be fruitful and the consequences are usually better than the traditional statistical methods. These facts lead to the close cooperation between machine learning and machine translation.

Nevertheless, there is no ML technology that can emulate human translation perfectly, leaving the investigation and development of ML methodologies for MT a challenging research field.

There exist numerous open questions when implementing ML techniques for MT. Overall, they fall in two main categories

1. How to build up a generalised model that can better characterise comprehensive features of human translations, such as morphology, syntax and semantics.

2. Meanwhile how to control the complexity of learning algorithms to adapt to the rapid growth of the available corpora database.

From the next chapter, we start presenting our work for MT, bearing these two open issues in mind when applying the cutting-edge ML technologies.

Chapter 4

Kernel–based machine translation

Symbol	Notation
$\mathbf{f}^i$	the i-th sentence (string) in the source set
$\mathbf{e}^i$	the i-th sentence (string) in the target set
N	the size of the source (target) set
$\mathbb{F}$	the input (source string) space
$\mathbb{E}$	the output (target string) space
$\mathcal{H}_\phi$	Hilbert space comprising the input feature vectors
ϕ	embedding function to map the input space to a feature space $\phi : \mathbb{F} \rightarrow \mathcal{H}_\phi$
$\mathcal{H}_\varphi$	Hilbert space comprising the output feature vectors
φ	embedding function to map the output space to a feature space $\varphi : \mathbb{E} \rightarrow \mathcal{H}_\varphi$
$\mathbf{W}$	a matrix-represented linear operator projecting $\mathcal{H}_\phi$ into $\mathcal{H}_\varphi$
$\mathbf{tr}(\mathbf{W})$	the trace of the matrix $\mathbf{W}$
$\|\mathbf{W}\|^2_{\mathcal{F}}$	the Frobenius norm of matrix-represented operator $\mathbf{W}$ and it is defined by $\|\mathbf{W}\|^2_{\mathcal{F}} = \mathbf{tr}(\mathbf{W}^T\mathbf{W})$
$K^\phi(\mathbf{f}^i, \mathbf{f}^j)$	the string kernel for the source strings $\mathbf{f}^i$ and $\mathbf{f}^j$, $K^\phi(\mathbf{f}^i, \mathbf{f}^j) = \langle \phi(\mathbf{f}^i), \phi(\mathbf{f}^j) \rangle$
$K^\varphi(\mathbf{e}^i, \mathbf{e}^j)$	the string kernel for the target strings $\mathbf{e}^i$ and $\mathbf{e}^j$, $K^\varphi(\mathbf{e}^i, \mathbf{e}^j) = \langle \varphi(\mathbf{e}^i), \varphi(\mathbf{e}^j) \rangle$
$\bullet$	element-wise product
A	the vocabulary of a language
Σ	a finite set of finite state automata (FSA) states

Table 4.1: Notations used in this chapter.

4.1 Introduction

As discussed in Section 2.5, the application of kernel–based analysis to MT is far less explored. To the best of our knowledge, so far only two kernel applications are related to MT: one is the string-to-string regression framework proposed by Cortes et al. (2005), the other is the least square regression used in (Wang et al., 2007). For the former, since its pre-image solution requires the strings coming from a regular language (e.g. finite state machine, tuning machine, etc), which cannot be proved true for human languages, it is unable to solve MT tasks directly. For the latter, although the framework can be applied directly to MT tasks, it shares too many components with the baseline SMT system (e.g. the language model and the beam search decoder), in which case the advantage of the proposed method is not clear.

In contrast, this chapter aims at developing a "pure" kernel–based MT system based on the regression framework used in (Cortes et al., 2005). The purpose of this study is to evaluate usability and reliability of kernel methods in the MT field. In contrast to ridge regression and n–gram kernels used by Cortes et al. (2005), a *maximum margin regression* framework (Szedmak et al., 2006) is introduced and two novel string kernels are utilised to model the bilingual relationship. Furthermore, existing weighting strategies for string kernels are exploited which leads to a general kernel learning approach. To meet the requirements of MT tasks, the pre-image solution in (Cortes et al., 2005) is also replaced by a Viterbi decoder, which generates the most probable string path as a target translation.

The following sections are organised as follows: in Section 4.2, we first describe the learning framework – maximum margin regression (MMR). Then two novel string kernels and a kernel learning framework are presented in Section 4.3. In addition, we introduce two data pre-processing techniques for constructing better kernels. Section 4.4 depicts the Viterbi decoder for the proposed MT system. The system is then evaluated in Section 4.5 using two French-to-English MT tasks and the results are compared with a traditional SMT system. Finally, in Section 4.6 we draw the conclusion and discuss the problems need to be solved.

4.2 Maximum margin regression

In a string-to-string regression framework, each source sentence $\mathbf{f}$ (or target sentence $\mathbf{e}$) is embedded into a linguistic feature space that has very high dimensions (usually $d \geq 50000$) but it is very sparse. The purpose is to predict the target feature vector, whose active (non-zero) features are then passed through a translation decoder to generate the target translation. In this sense, the task can be regarded as a multi-class problem that contains a vast amount of classes, or in an extreme case, infinite classes.

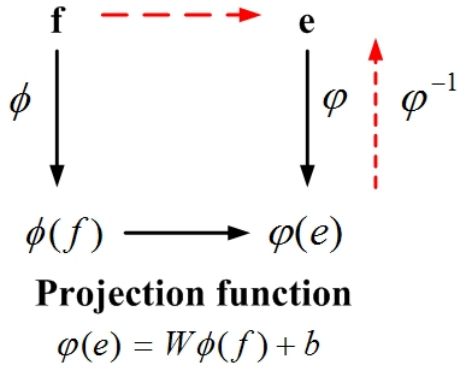


FIGURE 4.1: Illustration of the MMR phrase feature predictor. Both source ($\mathbf{f}$) and target ($\mathbf{e}$) sentences are embedded into their own feature spaces. Then MMR learns a linear operator $\mathbf{W}$ from the input feature space to the output feature space.

Since the number of output classes is considerably large, the traditional multi-class classification techniques such as multi-class SVMs are impractical to apply. As an alternative, Szedmak et al. (2006) introduced a novel approach, namely *maximum margin regression* (MMR), for large-scale multi-class or structured learning problems. This framework is ideal for learning langauge relations, because of its adaption to the dimension of the output vector: it is able to learn a vector output with the complexity of a binary SVM. Therefore, we make use of MMR as the core of our learning agent and develop the proposed MT system.

4.2.1 MMR phrase feature predictor

Figure 4.1 depicts the learning procedure of MMR. Mathematically, the source string $\mathbf{f} \in \mathbb{F}$ is embedded into a high–dimensional feature space with a function $\phi : \mathbb{F} \to \mathcal{H}_\phi$. Similarly the target string $\mathbf{e} \in \mathbb{E}$ is embedded into another feature space with $\varphi : \mathbb{E} \to \mathcal{H}_\varphi$. Then with existing sentence pairs $\mathcal{S} = \{(\mathbf{f}^i, \mathbf{e}^i) : \mathbf{f}^i \in \mathbb{F} \text{ and } \mathbf{e}^i \in \mathbb{E},\ i = 1, \ldots, N\}$, MMR learns a linear mapping $\mathbf{W} \in \mathbb{R}^{\mathcal{H}_\varphi \times \mathcal{H}_\phi}$ from $\mathcal{H}_\phi$ to $\mathcal{H}_\varphi$ by solving the following optimisation problem (Szedmak et al., 2006)

$$\begin{array}{ll} \min & \frac{1}{2}\mathbf{tr}(\mathbf{W}^T\mathbf{W}) + C\mathbf{1}^T\boldsymbol{\xi} \\ w.r.t & \{\mathbf{W}|\mathbf{W} : \mathcal{H}_\phi \to \mathcal{H}_\varphi, \mathbf{W} \text{ is a linear operator}\} \\ & \{\mathbf{b}|\ \mathbf{b} \in \mathcal{H}_\varphi,\ \mathbf{b} \text{ is a bias vector}\} \\ & \{\boldsymbol{\xi}|\boldsymbol{\xi} \in \mathbb{R}^N,\ \boldsymbol{\xi} \text{ is a slack or error vector}\} \\ s.t. & \langle\varphi(\mathbf{e}^i), (\mathbf{W}\phi(\mathbf{f}^i) + \mathbf{b})\rangle_{\mathcal{H}_\varphi} \geq 1 - \xi_i,\ i = 1, \ldots, N \\ & \boldsymbol{\xi} \geq \mathbf{0} \end{array} \tag{4.1}$$

where $\mathbf{0}$ and $\mathbf{1}$ denote the vectors with components 0 and 1 respectively.

Due to the high-dimensionality of the linguistic feature space, this primal formation usually suffers from a tremendous amount of weight parameters in $\mathbf{W}$. As discussion in

Section 3.4.2.1, a natural choice is to introduce dual variables $\boldsymbol{\alpha} = \{\alpha_i | i = 1, \ldots, N\}$ that can then be induced by solving the dual optimisation problem

$$\begin{array}{ll} \min & \sum\limits_{i,j=1}^{N} \alpha_i \alpha_j \langle \phi(\mathbf{f}^i), \phi(\mathbf{f}^j) \rangle \langle \varphi(\mathbf{e}^i), \varphi(\mathbf{e}^j) \rangle - \sum\limits_{i=1}^{N} \alpha_i \\ w.r.t & \{\alpha_i | \alpha_i \in \mathbb{R}\}_{i=1}^{N} \\ s.t & \sum\limits_{i=1}^{N} \alpha_i (\varphi(\mathbf{e}^i))_t = 0, \ t = 1, \ldots, dim(\mathcal{H}_\varphi) \\ & 0 \leq \alpha_i \leq C, i = 1, \ldots, N \end{array} \tag{4.2}$$

where the inner products in (4.2) are usually defined implicitly via the kernel functions

$$\begin{array}{c} K^{\phi}(\mathbf{f}^i, \mathbf{f}^j) = \langle \phi(\mathbf{f}^i), \phi(\mathbf{f}^j) \rangle \\ K^{\varphi}(\mathbf{e}^i, \mathbf{e}^j) = \langle \varphi(\mathbf{e}^i), \varphi(\mathbf{e}^j) \rangle \end{array} \tag{4.3}$$

4.2.2 Prediction and pre-image solution

Given the solution of $\boldsymbol{\alpha}$, the feature prediction of a target sentence $\mathbf{e}$ is of the form

$$\varphi(\mathbf{e}) = \sum_{i=1}^{N} \alpha_i \varphi(\mathbf{e}^i) K^{\phi}(\mathbf{f}, \mathbf{f}^i) \tag{4.4}$$

and the pre-image of the target sentence should be $\varphi^{-1}(\varphi(\mathbf{e}))$.

In general the direct inversion is infeasible. But it is possible to obtain an approximation by exhaustively searching the output space $\mathbb{E}$ such that

$$\begin{array}{rcl} \mathbf{e} & = & \arg\max\limits_{\mathbf{e}^t \in \mathbb{E}} \langle \varphi(\mathbf{e}^t), \mathbf{W}\phi(\mathbf{f}) \rangle_{\mathcal{H}_\varphi} \\ & = & \arg\max\limits_{\mathbf{e}^t \in \mathbb{E}} \sum\limits_{i=1}^{N} \alpha_i K^{\varphi}(\mathbf{e}^t, \mathbf{e}^i) K^{\phi}(\mathbf{f}, \mathbf{f}^i) \end{array} \tag{4.5}$$

Unfortunately, the target space $\mathbb{E}$ represents the target word sequences of arbitrary length, making the exhaustive search of space $\mathbb{E}$ intractable. This poses an implementation problem in finding the pre-image solution and approximate approaches are required to effectively decode the sentence. Alternative to the beam search decoder used in (Koehn, 2004; Wang et al., 2007), another approximation is utilised and discussed in Section 4.4.

4.3 Kernel application

There is no explicit feature expression in the MMR solution, as an alternative the kernels are imported to model the relations between sentences. Therefore, the kernels become

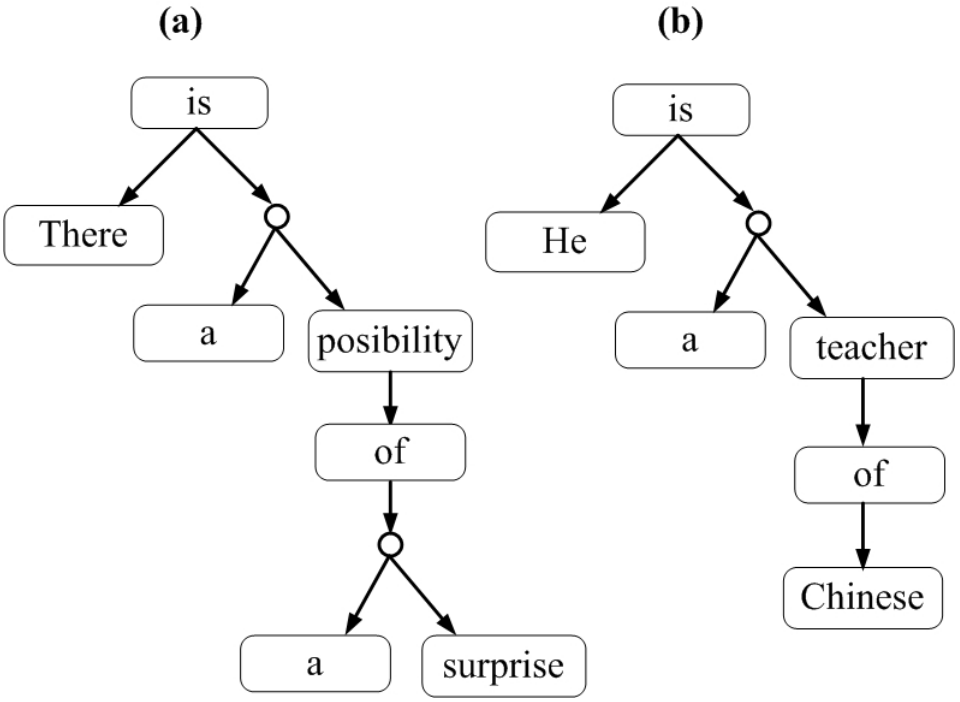


FIGURE 4.2: The parsing trees for the sentences "There is a possibility of a surprise" and "He is a teacher". The trees show the syntax structures of the two sentences, where the syntax blocks (represented by the subtrees) are in different length.

one of the most important component leading to the success of the MMR phrase feature predictor. Intuitively, the more relations between sentences are captured by its kernels, the better the MMR prediction will be.

The n–gram kernels form a big family of kernels for strings (Manning and Schütze, 1999), that measure the similarity between two strings using the count of common word sequences. Let $|s|_u$ denote the number of occurrences of string u in sentence s, the n–gram kernel K_n ($n \geq 1$) between two sentences s and z is defined by

$$K_n(s, z) = \sum_{|u|=n} |s|_u |z|_u$$

where the sum runs over all strings u of length n. In particular, when $n = 1$ the n-gram kernel measures the number of common words between s and z.

Although these kernels have been applied successfully in a variety of prediction tasks such as text categorisation (Shawe-Taylor and Cristianini, 2004), they have two weaknesses when applied to MT problems.

Firstly, they only count fixed-length substrings between two sentences. However, to capture a comprehensive source of sentence information such as content and syntax, substrings in different length should be considered. Take the sentences in Figure 4.2 for example, if a uni-gram ($n = 1$) kernel is applied to measure the similarity between

these sentences, the similarity of the syntax structure "is a" would be ignored. On the contrary, if a bi-gram ($n = 2$) kernel is considered, the preposition "of" is omitted.

Secondly, they uniformly weight all n–grams, but in practice, some n–grams are more informative than others. For instance, key words such as nouns (e.g. "teacher" in this example) and verbs are more informative than stop words (e.g. "a" in this example); longer n–grams are usually more informative than shorter ones.

To tackle these weaknesses, we introduce two novel kernels which have different points of view of informative blocks. Both kernels used in our work are at the word level and do not contain any syntax information. Observing that for different languages different types of kernels show the best performance, perhaps due to the respective patterns the kernels emphasise, we may use different mapping functions for the input and the output spaces (i.e. $\phi \neq \varphi$).

4.3.1 Finite state automata (FSA) kernel

A sentence s can be viewed as a set of overlapping n-grams $\{v\}$. If we regard each n-gram $v = ux$ as a transition from a state (a phrase or a word) $u \in A^{n-1}$ to a new state $v[2:n]$ which is controlled by a word $x \in A$, we arrive at a finite state automaton with states indexed by A^{n-1} and transitions labeled by the elements of A. In this way, a sentence s can also be represented in a transition feature space with several non-zero transition values, which is named *finite state automata* (FSA) space (Saunders et al., 2002).

Mathematically, let Σ denote a finite set of FSA states and $|s|$ denote the length of sentence s, then the probability of "generating" sentence s with FSA states is a product of values on all transitions used

$$P_F(s) = \prod_{j=1}^{|s|} p_{s[i_j:j-1] \to s_j} \tag{4.6}$$

where s_j denotes the j-th word in s and $i_j = \min\{i : s[i : j-1] \in \Sigma\}$. Figure 4.3 illustrates an example where Σ only consists of word states. The probability of generating s is computed by the product of transition probabilities through the red path.

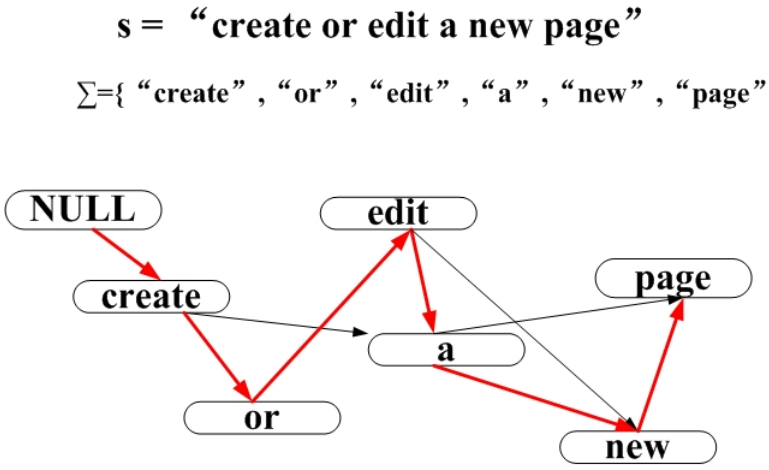


FIGURE 4.3: An example of generating a sentence based on FSA states. The red path shows the transition path of the sentence.

The derivative of the log–probability $\ln P_F(s)$ is then taken with respect to the variable $p_u(x)$ and yields

$$\begin{aligned} \frac{\partial \ln P_F(s)}{\partial p_u(x)} &= \frac{\partial \ln \prod_{j=1}^{|s|} p_{s[i_j:j-1]\to s_j}}{\partial p_u(x)} \\ &= \sum_{j=1}^{|s|} \frac{\partial \ln p_{s[i_j:j-1]\to s_j}}{\partial p_u(x)} \\ &= \frac{|s|_{(u,x)}}{p_u(x)} \end{aligned} \tag{4.7}$$

where $|s|_{(u,x)}$ denotes the frequency of (u, x) in sentence s. This expression is then defined as the transition feature from u to x for sentence s

$$\phi_{u,x}(s) = \frac{|s|_{(u,x)}}{p_u(x)} \tag{4.8}$$

and the FSA kernel is subsequently computed by

$$K^{FSA}(s, z) = \langle \phi(s), \phi(z) \rangle_{\mathcal{H}_\Sigma} = \sum_{u\in\Sigma, x\in A} \frac{|s|_{(u,x)}|z|_{(u,x)}}{p_u^2(x)} \tag{4.9}$$

If $p_u(x)$ is set as a uniform distribution (i.e. $p_u(x) = \frac{1}{|\Sigma||A|}$), implying that all features are equally informative, the FSA kernel is reduced to a joint n-gram kernel. To emphasise more informative features (transitions) and discount those otherwise, the rare features that occur less than a threshold t are pruned and $p_u(x)$ is given by the formula (Saunders et al., 2002)

$$p_u(x) = \frac{f_{u,x} + c}{|A|c + \sum_{x'\in A} f_{u,x'}} \tag{4.10}$$

in which $f_{u,x}$ is the number of occurrences we leave state u to process symbol x (i.e., the

frequency of n–gram ux) in the whole data set, A is the vocabulary and c is a pseudo count. Given a proper value of c, the distribution (4.10) would assign large values to frequent transitions (n–grams) and small values to rare ones, which reflects its preference for frequent transitions.

4.3.1.1 Modified FSA kernel

To derive a good prediction, a well-defined kernel is essential. Since we do not explicitly represent the sentences but only design a kernel to model sentence similarities, the kernel should collect as much similarity information as possible. This information is represented by the "density" of the kernel. Although the FSA model is possible to capture informative blocks of sentences, usually it is too sparse ($\leq 0.01\%$ of kernel values are non-zero). To broaden the informative blocks and enhance the similarities between sentences, the condition $i_j = \min\{i : s[i : j-1] \in \Sigma\}$ used in equation (4.6) is replaced by

$$i_j = \{i : s[i : j-1] \in \Sigma\} \tag{4.11}$$

Expression (4.11) means for each transition on word j, we count all possible states in s which process s_j afterwards. In other words, we combine all possible paths "generating" sentence s and the probability is then a product of all path probabilities

$$P_{MF}(s) = \prod_{j=1}^{|s|} \prod_{i_j} p_{s[i_j:j-1] \to s_j} = (P_F(s))^{N_s}$$

with N_s denoting the number of possible paths "generating" sentence s.

Following a similar derivation of equation (4.7), we obtain a new feature vector $\tilde{\phi}_{u,x}(s) = \frac{|s|_{(u,x)}}{p_u(x)}$ which contains all possible transition blocks for sentence s. Then the modified FSA kernel is computed in the same way as equation (4.9).

4.3.2 Common substrings (CS) kernel

Instead of counting fixed-length blocks of two sentences, the *common substrings* (CS) kernel compares the set of all common substrings (n–grams) between two sentences. This kernel was first introduced in (Shawe-Taylor and Cristianini, 2004) but it requires polynomial operations. The complexity is then reduced to linear time by Vishwanathan and Smola (2004), using annotated suffix trees. Recently, Vishwanathan and Teo (2006) replaced suffix trees with suffix arrays and further reduced the time and space needed for the kernel computation, making the CS kernel applicable to large data sets.

The motivation of utilising the CS kernel is to explore different length common substrings and exploit those informative. The original CS kernel is on the basis of character level,

but it is easy to transplant this idea to word level so as to match the n–gram word representation.

Definition 4.1. (**Common substrings kernel**) Let $|s|_u$ denote the frequency of n–gram string u ($n \geq 1$) in a sentence s, the family of *common substrings kernels* is of the form (Vishwanathan and Teo, 2006)

$$\begin{aligned} K(s,z) &= \sum_{u \in s, u' \in z} w_u \delta_{u,u'} \\ &= \sum_{n=1}^{\infty} \sum_{|u|=n} w_u |s|_u |z|_u \end{aligned} \tag{4.12}$$

That is, the kernel counts the frequency of every substring (n–gram) u in both s and z and weighted it by w_u, which is used to favour more informative n–grams. To simplify the model, w_u is set as the length of u (i.e. $w_u = |u|$) so as to favour longer n–grams, but it is also possible to be tuned by cross-validation. The readers are referred to (Vishwanathan and Teo, 2006) for the computation of a CS kernel.

4.3.3 Relationships among n–gram kernel, FSA kernel and CS kernel

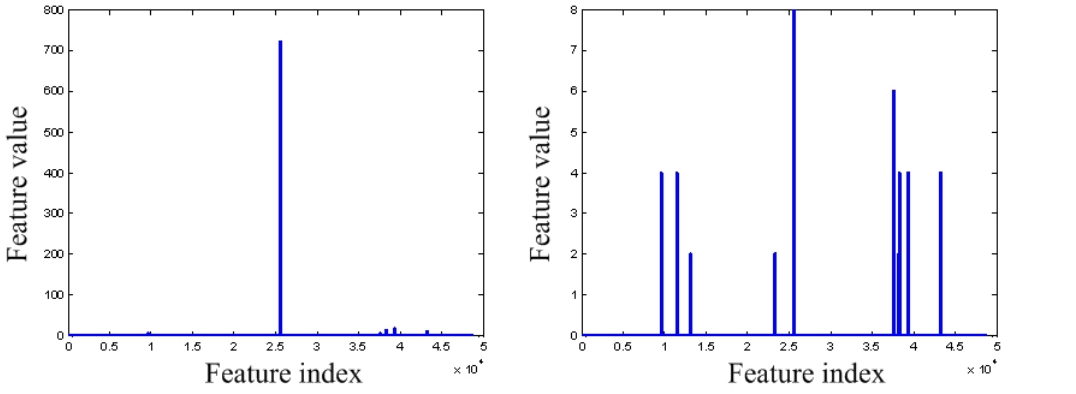
From the definitions, both the FSA and the CS kernels can be viewed as variants of n–gram kernels, with different weighting strategies. They attempt to tackle the weaknesses of n-gram kernels caused by fixed-length and uniformly weighting and are expected to capture more informative message blocks from the sentences.

In particular, we observed that if ignoring the pruning of rare features, the modified FSA kernel is exactly a CS Kernel with a specific weight w_u. The proof is easy to achieve. When constructing the modified FSA features we count all transitions $u \rightarrow x$ each of which corresponds to one unique n–gram (u, x). In other words, we count every n–gram with a weight $\frac{1}{p_u(x)}$, which is equivalent to a CS Kernel with the weight $w_{u,x} = \frac{1}{p_u(x)}$.

4.3.4 Advantages and weaknesses of FSA and CS kernels

The weighting strategy of FSA kernels depends on the frequencies of transitions. For example, when $|A|c$ is very large ($|A|c \gg \sum_{x' \in A} f_{u,x'}$), the weight $w_{u,x} = \frac{1}{p_u(x)}$ is inversely proportional to the frequency of ngram $\{u, x\}$, hence FSA favours very rare phrases (e.g. very long n-grams). On the contrary, if $|A|c$ is small the weight $w_{u,x}$ biases towards very rare transitions (i.e. u appears very frequently but x appears very rarely). Alternatively, the weighting strategy of CS Kernels simply favours longer n–grams.

Figure 4.4 shows the feature vectors for an English sentence “to enable the network agent service on the digipath workstation complete the following steps on the digipath



(a) FSA feature vector (b) Common Substring kernel feature vector

FIGURE 4.4: The FSA and the CS feature vectors for the English sentence "to enable the network agent service on the digipath workstation complete the following steps on the digipath workstation". Values in the CS feature vector are smoother than those in the FSA feature vector. The FSA feature with the highest value (the peak) is "on the digipath".

workstation"[1], constructed by FSA and CS models respectively. We observed that the features of the FSA model are much shaper than the CS features. As a result, if two sentences share a very rare transition, a FSA kernel would assign a very high similarity between them. This finally affects the MMR prediction that might only predict the rare transitions correctly.

Alternatively, the CS kernel combines all length of informative blocks and is expected to achieve the best performance among these kernels. However, in some cases it performed badly. The reason is that it assigns very high similarities between the same sentences, making the kernel matrix a diagonally dominant matrix. In this case, the normalised CS kernel matrix is just an identity matrix which contains no similarity information between sentences.

4.3.5 Learning a kernel

To eliminate or reduce the above weaknesses and improve the performance, we introduce a kernel learning framework – *Multiview learning* (Szedmak et al., 2006)

Proposition 4.2. *Suppose there are several sources of inputs* $\{\varphi(\mathbf{e}^i), (\phi^1(\mathbf{f}^i), \phi^2(\mathbf{f}^i), ...)\}$ *taken out of distinct distributions, then multiview learning solves the following optimisation problem*

[1]The sentence comes from the Xerox copy manual corpus, which is described in Section 4.5.1

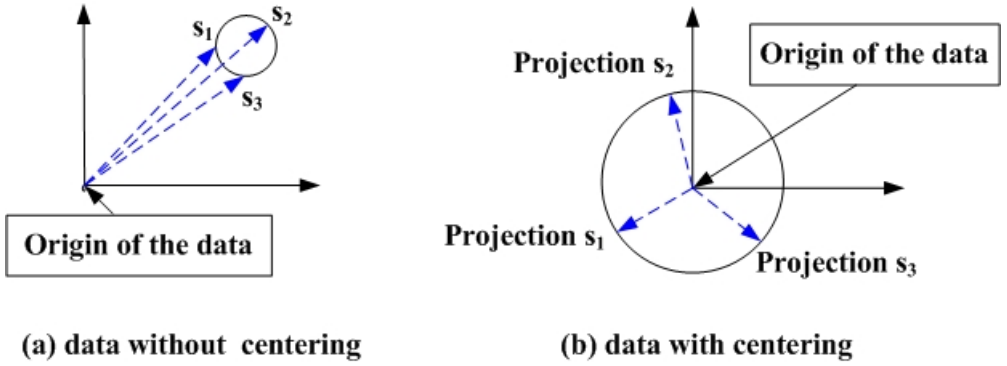


FIGURE 4.5: The effect of centering the data.

$$
\begin{aligned}
\min \quad & \tfrac{1}{2}\sum_{k=1}^{m} \boldsymbol{tr}(\mathbf{W}_k^T\mathbf{W}_k) + C\mathbf{1}^T\boldsymbol{\xi} \\
w.r.t \quad & \mathbf{W}_k : \mathcal{H}_{\phi^k} \rightarrow \mathcal{H}_{\varphi},\ \textit{linear operators} \\
& \boldsymbol{b} \in \mathcal{H}_{\varphi},\ \textit{a bias vector} \\
& \boldsymbol{\xi} \in \mathbb{R}^N,\ \textit{a slack or error vector} \\
s.t. \quad & \langle \varphi(\mathbf{e}^i), \sum_{k=1}^{m} \mathbf{W}_k\phi^k(\mathbf{f}^i) + \boldsymbol{b}\rangle_{\mathcal{H}_{\varphi}} \geq 1 - \xi_i \\
& \xi_i \geq 0, i = 1, \ldots, N
\end{aligned}
$$

which is equivalent to solving problem (4.2) with the corresponding Multiview kernel $\boldsymbol{K}^{\varphi} \bullet \sum_{k=1}^{m} \boldsymbol{K}^{\phi^k}$ *with* $\bullet$ *denoting the element-wise product.*

This proposition allows us to extract sentence similarities by designing a Multiview kernel. For example, it is easy to show that the CS kernel (4.12) is an infinite view of n–gram kernels, with a set of weights $\{w_u : w_u = |u|\}$. Motivated by the structured classification technique, the weights can also be set as $\{w_u : w_u = q^{|u|}\}$ with $q > 1$ favouring longer n-grams. As observed in the MT experiments, this Multiview kernel with an optimal q selected by cross-validation achieved an improved performance over the other kernels.

Moreover, it is likely to combine different types of kernels such as the CS and the FSA kernels to construct a more robust string kernel, in which case previous works (Joachims et al., 2001; Gao et al., 2002; Argyriou et al., 2006) can provide helpful paradigms. The further investigation of Multiview kernels is left to our future work.

4.3.6 Data pre–processing – centering and normalisation

To achieve an MMR prediction, two steps of data pre-processing are required apart from the well-designed kernels: centering (Figure 4.5) and normalisation (Figure 4.6).

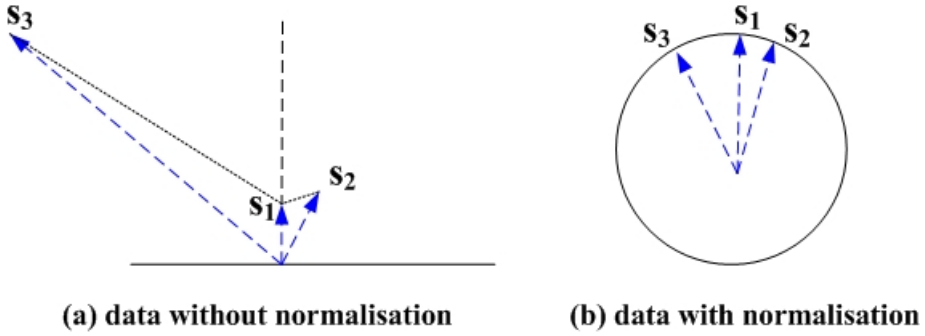


FIGURE 4.6: The effect of normalising the data.

The aim of centering is to move the origin of the feature space to the center of mass of the data. As shown in (Meilă, 2002), if in a feature space the origin lies far away from the data (Figure 4.5 (a)), then the kernel matrix $\mathbf{K}$ will have almost equal elements and is ill–conditioned. To avoid this problem, a standard *centering* method is usually applied in data pre–processing. Since the maximum margin hyperplane is invariant to translations in feature space, the centering technique would not affect the resulting classifier.

Working in a kernel–based feature space means that we are not able to explicitly represent the sentences, and one effect that might cause problems is the influence of the length of a sentence. Clearly, the longer a sentence is the more words it contains, and hence the greater the norm of its associated vector is. Take a toy kernel for example, assume that there are three sentences s_1="a warning message appears", s_2="a warning message is displayed" and s_3="you will get a warning message when you attempt to transform a double into an int, while you can stop the warning message by creating a warning elimination", each of which is represented by bi–gram ($n = 2$) features. Without normalisation the bi–gram kernel values are $K(s_1, s_2) = 2$ and $K(s_1, s_3) = 4$ (Figure 4.6 (a)). This indicates s_1 and s_3 are more similar, while in practice s_1 and s_2 should be closer, either in semantics or syntax. Alternatively, after normalisation the effect of sentence length is removed and we obtain $\bar{K}(s_1, s_2) = 0.577$ and $\bar{K}(s_1, s_3) = 0.436$, which matches the practice (Figure 4.6 (b)). As the length of a sentence is not relevant for representing the similarities between sentences, it is reasonable to remove this effect by normalising the embedding vectors.

Both pre-processings can be applied to the kernels directly, which avoids the operations on the feature space. Mathematically, let ϕ_{S_f} to be the center of the source sentences $\{\mathbf{f}^i\}_{i=1}^N$

$$\phi_{S_\mathbf{f}} = \frac{1}{N}\sum_{i=1}^{N}\phi(\mathbf{f}^i) \tag{4.13}$$

the new feature vector for sentence $\mathbf{f}^i$ is computed by

$$\hat{\phi}(\mathbf{f}^i) = \phi(\mathbf{f}^i) - \phi_{S_\mathbf{f}} \tag{4.14}$$

and the source kernel is centered as

$$\begin{aligned} \hat{K}^\phi(\mathbf{f}, \tilde{\mathbf{f}}) &= \langle \hat{\phi}(\mathbf{f}), \hat{\phi}(\tilde{\mathbf{f}}) \rangle_{\mathcal{H}_\phi} \\ &= K^\phi(\mathbf{f}, \tilde{\mathbf{f}}) - \tfrac{1}{N} \sum_{i=1}^{N} K^\phi(\mathbf{f}, \mathbf{f}^i) - \tfrac{1}{N} \sum_{i=1}^{N} K^\phi(\tilde{\mathbf{f}}, \mathbf{f}^i) + \tfrac{1}{N^2} \sum_{i,j=1}^{N} K^\phi(\mathbf{f}^i, \mathbf{f}^j) \end{aligned} \tag{4.15}$$

After centering, the normalisation gives

$$\bar{\phi}(\mathbf{f}) = \frac{\hat{\phi}(\mathbf{f})}{\|\hat{\phi}(\mathbf{f})\|} \tag{4.16}$$

that yields the normalised kernel

$$\bar{K}^\phi(\mathbf{f}, \tilde{\mathbf{f}}) = \frac{\hat{K}^\phi(\mathbf{f}, \tilde{\mathbf{f}})}{\sqrt{\hat{K}^\phi(\mathbf{f}, \mathbf{f}) \hat{K}^\phi(\tilde{\mathbf{f}}, \tilde{\mathbf{f}})}} \tag{4.17}$$

The same operations are then applied to the target side to construct the target centered and normalised kernel.

4.4 Pre-image solution – MMR with Viterbi

An MT system usually involves two stages: *training* and *decoding*. This section presents the development of an appropriate decoder for the kernel–based MT system, which is inspired by the beam search decoder[2] used in current SMT systems.

A traditional phrase–based SMT model trains a phrase translation table (Figure 4.7) before decoding, where easier translated phrase pairs are assigned larger probabilities. In general, the translation table can also be viewed as a common substrings feature space, with an ordered one-to-one relationship between the input and output features (phrases). To construct the target feature vector $\varphi(\mathbf{e})$, only the target features whose corresponding sources appear in $\mathbf{f}$ are assigned the probabilities, while others have zero values. During the decoding, the model picks up those non-zero target features and builds up the translation from left to right. This decoder actually searches the approximately highest probability path for the output translation, which is a variant of Viterbi decoding at the phrase level.

[2]The readers are referred to Section 2.4.6 for the review of the beam search decoder.

documentation est includes dans le	**#**	**documentation is provided in the**	**#**	**1**
documentation est includes dans	**#**	**documentation is provided in**	**#**	**1**
documentation est includes	**#**	**documentation is provided**	**#**	**1**
documentation est	**#**	**documentation is**	**#**	**1**
la documentation est includes dans	**#**	**the documentation is provided in**	**#**	**1**
la documentation est includes	**#**	**the documentation is provided**	**#**	**1**
la documentation est	**#**	**the documentation is**	**#**	**1**
est fourni avec le	**#**	**is included with your**	**#**	**0.5**
est fourni avec le	**#**	**is provided within the**	**#**	**0.5**
est fourni avec	**#**	**is included with**	**#**	**0.4**
est fourni avec	**#**	**is provided within**	**#**	**0.4**

FIGURE 4.7: Part of the French-to-English translation table generated by a phrase–based SMT system. There are three items displayed: the source phrase (French), the target translation (English) and the probability.

4.4.1 MMR with the Viterbi decoder

Alternatively, the MMR prediction operates at the sentence level. Given $\mathbf{f}$, the MMR model searches the sentence space $\mathbb{E}$ for a translation that maximises the objective function (4.5). In general the phrase space is a sub-space of the sentence space, hence searching in space $\mathbb{E}$ is much more time-consuming than searching in a phrase space. In this respect, the decoder for the MMR phrase feature predictor confronts a more difficult problem than a traditional SMT decoder encounters.

However, by reverting the target kernel back to feature space, the decoder for MMR can operate at the phrase level as well. Recall that the output vector of MMR is

$$\bar{\varphi}(\mathbf{e}) = \sum_{i=1}^{N} \alpha_i \bar{\varphi}(\mathbf{e}^i) \bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i)$$

with $\bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i)$ representing the similarity between sentences $\mathbf{f}$ and $\mathbf{f}^i$. As natural languages, if $\bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i)$ is large, that is, $\mathbf{f}$ and $\mathbf{f}^i$ share some source features (i.e. common substrings), $\mathbf{e}$ and $\mathbf{e}^i$ should have some target features in common as well. Under this assumption, by comparing $\mathbf{f}$ with all sentences in $\mathcal{S}$, the target features in $\bar{\varphi}(\mathbf{e})$ are expected to have non-zero values (at least those appeared in $\mathcal{S}$), where more important features would receive larger weights. Then it is reasonable to view $\bar{\varphi}_u(\mathbf{e})$ as the "probability" to emit feature (target phrase) u for sentence $\mathbf{e}$. In this way, the pre-image problem is at the phrase level and the goal is to combine as many high "probability" features $\{u\}$ as possible.

Inspired by beam search, the Viterbi algorithm seems to be a perfect fit for the decoder. The pseudo-code is described in Table 4.2.

Input: states (feature vector) $\{u_i\}_{i=1}^{dim(\varphi)}$, "probabilities" to emit states $\{P_{\mathbf{e}}(u_i)\}_{i=1}^{dim(\varphi)}$, transition probabilities from state i to state j $\{P_{tr}(u_i, u_j)\}$ and number of states τ.

1) **Initialise** the probability $\delta_0(u_i) = P_{\mathbf{e}}(u_i)$

2) **for** $m = 1 \cdots \tau$ **do**

   a) $\delta_m(u_i) = \max_{j=1}^{|s|}\{\delta_{m-1}(u_j)P_{tr}(u_j, u_i)\}P_{\mathbf{e}}(u_i)$

   b) $\text{index}_m(u_i) = \arg\max_{u_j} \delta_{m-1}(u_j)P_{tr}(u_j, u_i)$

   end for

3) Derive the highest probability $P_{max} = \max_{u_i}\{\delta_\tau(u_i)\}$

4) Find the Viterbi path $\text{Path}_\tau = \arg\max_{u_i}\{\delta_\tau(u_i)\}$

   for $m = \tau \cdots 2$ **do**

   $\text{Path}_{m-1} = \text{index}_m(\text{Path}_m)$

   end for

Output: Viterbi path Path and its probability P_{max}.

TABLE 4.2: Pseudo-code of the Viterbi decoder.

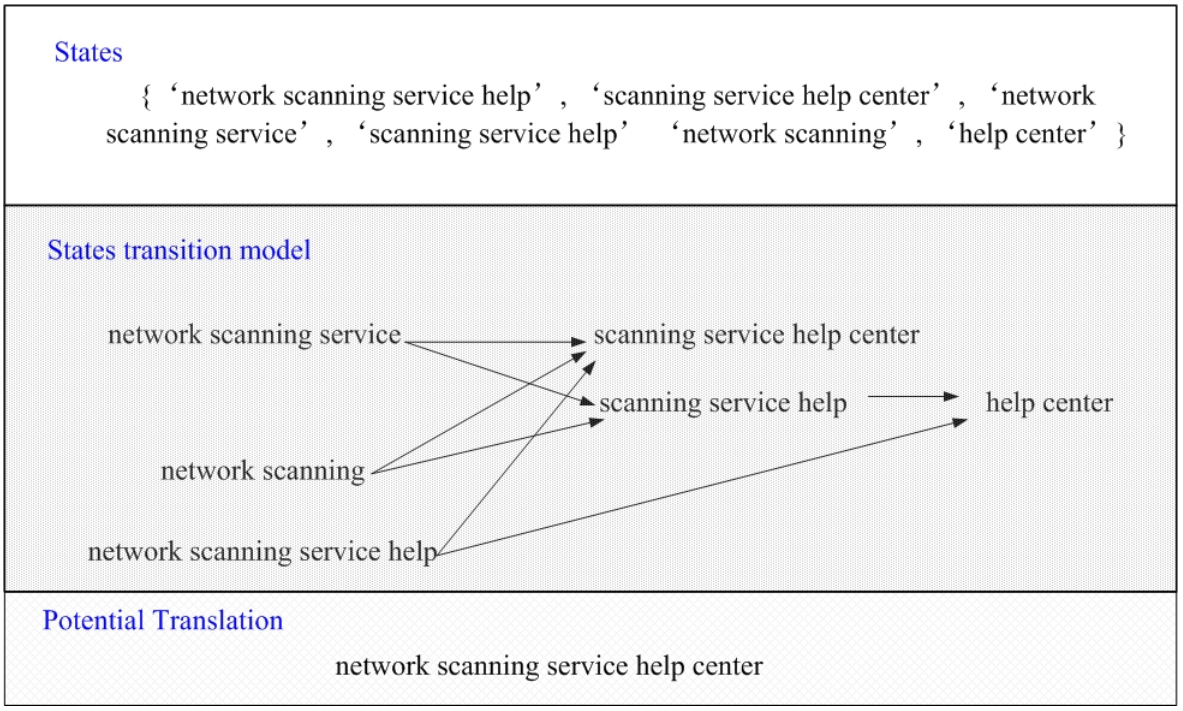


FIGURE 4.8: The overlap transition. One state can only transit to another that has overlapping words.

Mathematically, the feature–emission probability is a function of the MMR prediction

$$P_{\mathbf{e}}(u_i) = \frac{1}{Z}\exp\{\bar{\varphi}_{u_i}(\mathbf{e})\} \tag{4.18}$$

with the partition function $Z = \sum_{u_j}\exp\{\bar{\varphi}_{u_j}(\mathbf{e})\}$.

To further improve the fluency of output translation, an overlapping transition probability model is also constructed. Figure 4.8 illustrates an example. A state can only transit to another if and only if its suffix is another's prefix and the transition probability is then computed by

$$P_{tr}(u_i, u_j) = \Big(\frac{f_{u_i,u_j}}{\sum_{j'} f_{u_i,u_{j'}}}\Big)^{\lambda} \tag{4.19}$$

where f_{u_i,u_j} is the transition frequency from u_i to u_j and λ is a scale factor to control the influence of the transition probabilities.

With this auxiliary model, the abnormal transitions (e.g. "from the menu→select layout") would receive very low possibilities where in contrast the fluent ones would have high possibilities. In this way, the Viterbi decoder is expected to pick up "high probability" features and combining them in a fluent way.

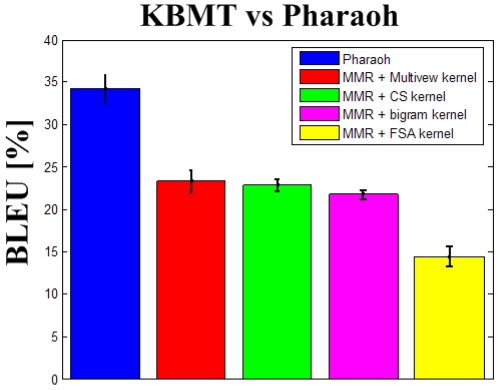
4.5 Translation experiments

4.5.1 Experiment setup

The corpus used for the MT experiments is the Xerox copy manual corpus[3], where the task is to translate the *French* sentences into *English*. For scaling up purpose, a split of the corpus with $10,000$ sentences (length between 6 and 15) was randomly sampled for training and other 800 sentences were chosen for evaluation. Although different size feature vectors were constructed for different kernels, the features were always joint n-grams ($n \geq 1$) and on average the feature vector contained more than $100,000$ items. Overall there were $28,849$ target features in the test set, of which only 36.03% appeared in the training domain. This situation makes the feature prediction a difficult task.

To compare the performance, a traditional SMT system, Pharaoh (Koehn, 2004), was used as the baseline system. In addition, we also compared the performance among four kernels for MMR: the *Multiview* kernel, the *CS* kernel, the *FSA* kernel and the *n-gram* kernel.

As commonly used in the MT evaluation, the BLEU score (Papineni et al., 2002) was utilised to measure the translation performance, and the experiments were repeated three times to assess variance.



Pharaoh	34.2 ± 1.6
MMR + Multiview kernel	23.3 ± 1.2
MMR + CS kernel	22.9 ± 0.7
MMR + bigram kernel	21.8 ± 0.5
MMR + FSA kernel	14.4 ± 1.2

FIGURE 4.9: The BLEU scores for the French-to-English MT task, using Pharaoh and the kernel–based MT system respectively. The error bars indicate the variance.

4.5.2 Translation results

Figure 4.9 displays the BLEU scores for different systems. As expected, the Multiview kernel shows a great flexibility in emphasising the informative phrase blocks and produces improved performance over the other kernels. The FSA kernel shows the worst performance, possibly due to the problem mentioned in Section 4.3.4; and we postulate it would work better in text categorisation rather than machine translation. Unfortunately, in none of the cases does the kernel-based MT system surpass Pharaoh.

4.5.3 Advantages and weaknesses of MMR with Viterbi

Figure 4.10 lists 10 sentences translated by Pharaoh and the kernel–based MT system, from which we conclude some advantages and weaknesses of the proposed system.

From the translation samples we observed that MMR with Viterbi is able to consider phrase reordering automatically using its overlapping transition model (see Figure 4.11), which is an advantage over the traditional SMT system that has to employ a language model to help the fluency of translation. However, its disadvantages are also observed. One disadvantage is illustrated in Figure 4.12. If some features rarely occur in the training set, the corresponding predictions for these features would have very low values

[3]Data supplied by XRCE Europe, which contains 25 technical documents and user guides. There are 53,568 sentence pairs (English and French) in the corpus, many of which are short sentences. The average sentence lengths of the corpus are 10 words (English) and 11 words (French).

Output by KBMT	**Reference**	**Output by Pharaoh**
1. config users xst w r w r	1. config drdf xst w r w r (**never** happen in the training set)	1. config dldf xst w r w r
2. installing to the default directory is recommended	2. installing to the default directory is recommended (happen **3 times** in the training set)	2. it is recommended deffectuer installation to the default directory
3. create or edit a distribution template	3. create or edit a distribution template (happen **once** in the training set)	3. creating or amending a distribution template
4. the manager user templates dialog opens	4. the manager user templates dialog opens (happen **3 times** in the training set)	4. the dialog manage user templates is displayed
5. 2 select layout from the print options menu	5. 2 select layout from the print options menu (happen **twice** in the training set)	5. 2 from the print options menu select layout
6. select manage user templates 4	6. manage user templates 3 14 ("manage user templates" happen **5 times** in the training set)	6. manage user templates 3 14
7. centreware network scanning services on netware	7. installing network scanning service 4 15 ("installing network scanning service" happen **once**)	7. installing network scanning service 4 15
8. introducing centreware printer drivers for macintosh environments	8. introducing centreware printer drivers for macintosh v ("introducing centreware printer drives for macintosh" happen **once**)	8. introducing centreware printer drivers for macintosh v
9. fax from the job type list box select secure print	9. 5 from the job type list box select sample set ("from the job type list box select" happen **once**)	9. 4 from the list job type select sample set
10. a select start programs xerox centreware fax phone book	10. select start programs xerox centreware device admin wizard ("xerox centreware device admin wizard" happen **once**)	10. select start programs xerox centreware device admin wizard

FIGURE 4.10: Translations (English) generated by Pharaoh and the kernel–based MT system. Left: sentences translated by the kernel-based MT system; Middle: Reference (English); Right: sentences translated by Pharaoh.

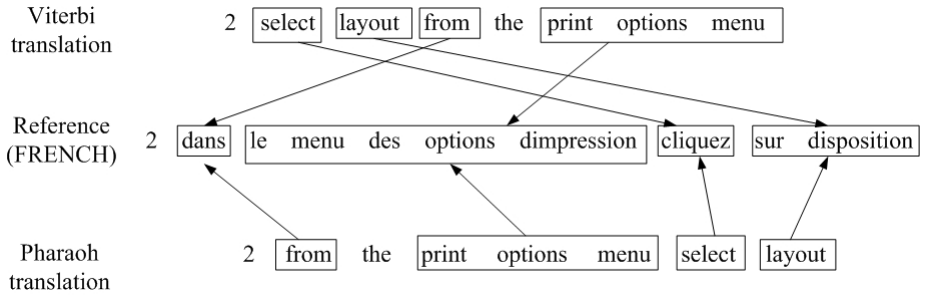


FIGURE 4.11: Example: compare with Pharaoh, MMR with Viterbi can generate more fluent translations.

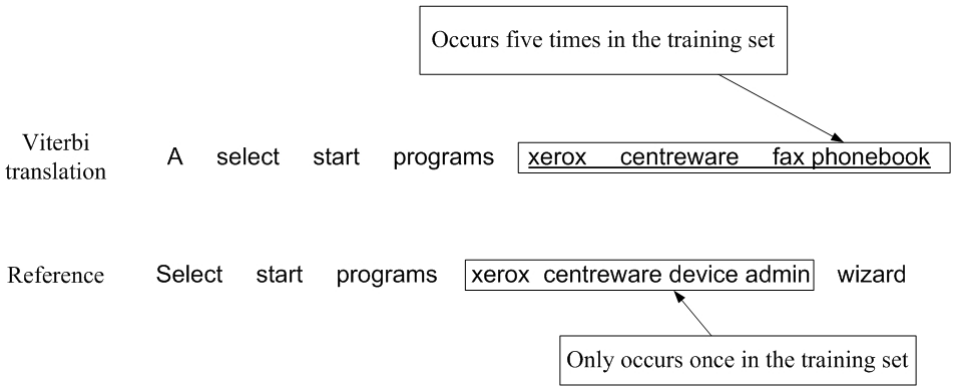


FIGURE 4.12: Incorrect path generated by the Viterbi decoder. Since the phrase "xerox centreware device admin" only occurs once in the training set, MMR has very low predicted values for the features like "centreware device admin" and "xerox centreware device". In this case, the Viterbi decoder finds it difficult to pick up the correct path "xerox center device admin wizard".

and then they are omitted by the Viterbi decoder. We postulate that this is due to the implicit expression of the sentences. As given by equation (4.4), the MMR prediction is

$$\bar{\varphi}(\mathbf{e}) = \sum_{i=1}^{N} \alpha_i \bar{\varphi}(\mathbf{e}^i) \bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i)$$

That is, for two source sentences with common substrings ($\bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i) > 0$), all features in the target sentence $\mathbf{e}^i$ (i.e. $\bar{\varphi}(\mathbf{e}^i)$) are updated, irrespective of whether they are related to the common parts or not. Therefore, the final prediction contains not only the information related to the target translation but also a lot of unexpected phrase features. In this case, the rare features might be overwhelmed by these unexpected features. Since many features in a sentence are rare features (represented by the tremendous amount of rare features in Figure 4.13), the MMR prediction (4.4) makes it difficult for the Viterbi decoder to generate quality translations.

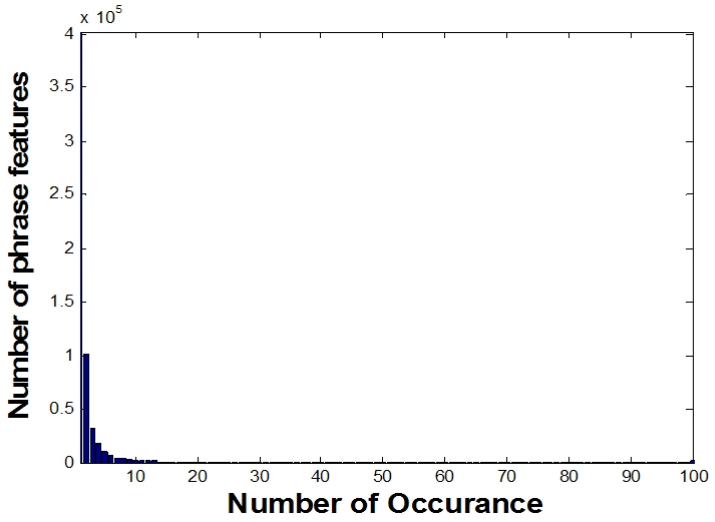


FIGURE 4.13: The statistics of frequencies of features (phrases) extracted from the Xerox copy manual corpus (English domain). The figure shows that many features (83.8% of the whole features) are rare features, which only occur less than three times in the corpus.

4.5.4 Improving MMR prediction – incorporating word alignments

One way of improving MMR predictions is to eliminate or reduce the amount of unexpected features. In other words, it is better to update only the target features that are related to the common source substrings. A straightforward solution is to incorporate word alignments between source sentences and target translations. Figure 4.14 illustrates an example. Given the word alignments for $(\mathbf{f}^i, \mathbf{e}^i)$, searching target features related to common source parts is as easy as looking up aligned translations for common source words. Hence we can replace the $\mathbf{e}^i$ term in equation (4.4) by the translation of the common source part, $\mathbf{u}^i$, to give

$$\bar{\varphi}(\mathbf{e}) = \sum_{i=1}^{N} \alpha_i \bar{\varphi}(\mathbf{u}_i) \bar{K}^{\phi}(\mathbf{f}, \mathbf{f}^i), \tag{4.20}$$

where $\mathbf{u}^i = \{u : u \subset \mathbf{e}^i \text{ and } (u, \mathbf{f} \wedge \mathbf{f}^i) \in \text{Alignment Table}\}$.

To verify the effectiveness of this MMR representation, another French–to–English MT task was set up. The data set was still a split of the Xerox copy manual corpus that contained $20,500$ training sentences (length between 5 and 40) and 848 test sentences. A Multiview kernel was used and the word alignments generated by a word alignment toolkit, GIZA++ (Och et al., 1999), were utilised to refine the MMR predictions. Figure 4.15 shows the translation results. The MMR predictor incorporated with the word alignments produces an absolute 4.1% improvement over the original one, which is disappointingly worse than Pharaoh again. There are two possible reasons for this. Firstly, the sentence level feature vector (e.g. $\bar{\phi}(\mathbf{f}^i)$) makes the phrase predictions inaccurate, due to the difficulty in designing a kernel that balances a batch of phrases perfectly. For

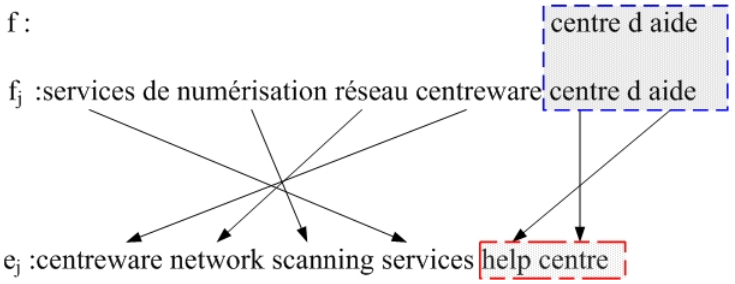


FIGURE 4.14: Incorporating the word alignments to update the target features, which are related to the common source features only. In this example, the word alignments are produced by the word–alignment toolkit $GIZA++$.

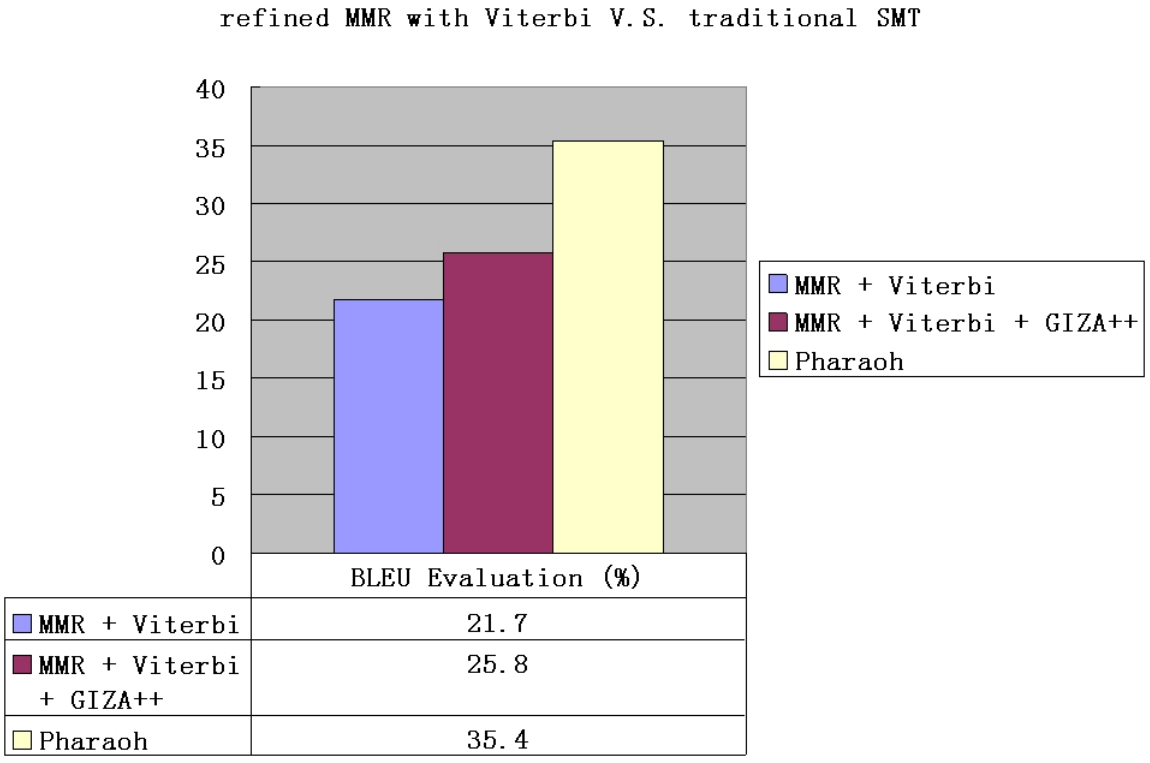


FIGURE 4.15: The BLEU scores for the French-to-English MT task.

example, the FSA model gives good predictions for rare phrases while other phrases are totally ignored. On the contrary, the CS model has smoother feature values. But when it is applied, very frequent phrases (e.g. "the", "that") dominate the prediction, making it difficult for the Viterbi decoder to pick up rare phrases.

Secondly, the Viterbi decoder might not be powerful enough to generate fluent translations. Hence, how to integrate a language model into the proposed system, or how to improve this existing decoder to have an equal performance is a further challenge for our future work.

4.6 Conclusion

The kernel–based MT system presented in this chapter consists of two components: an MMR phrase feature predictor and a Viterbi decoder. The overall performance is dominated by the joint performance of the MMR phrase prediction and the Viterbi overlapping transition model. Although the preliminary MT results do not outperform the baseline system, the time efficiency MMR provides (as discussed in Section 3.4.2.1), and the quality of fluency the Viterbi decoder achieves are still encouraging. In addition, incorporating the source–target word alignments so as to refine the MMR predictions is sound, which led to a promising improvement in an French-to-English translation experiment.

We believe there to be two specific problems in the proposed MT system. One is how to scale up to large data sets. Currently the kernels constructed in the preliminary experiments are for $20,000$ sentences, whereas some MT data collections (e.g. the EuroParl corpora) contain more than one million sentences. As the space complexity of a kernel of N samples is $O(N^2)$, the kernel matrix is too expensive to store when the data size is large. Theoretically, a compromise can be made by solving the MMR problem via re-constructing a small kernel for each sample involved, in which case the space complexity is reduced to $O(N)$. However, the time complexity of the MMR solution would increase from $O(N)$ to $O\big(N\big(dim(H_\phi) + dim(H_\varphi)\big)\big)$ for each inner loop in *sequential minimal optimisation* (SMO). This memory-to-runtime exchange could make the time complexity difficult to handle, especially when the dimensionality of the feature space is high.

The other problem is the accuracy of MMR predictions. The experiments have shown that the quality of a Viterbi path depends on the accuracy of MMR predictions and the performance of the overlapping transition model. Therefore, improving the accuracy of MMR predictions can also improve the quality of output translation. However, current MMR predictions still need improvement, even when incorporated with the source–target word alignments. Since MMR is a kernel–based ML method, a more powerful and robust kernel might be helpful, while the design of such a kernel for predicting a batch of phrases simultaneously is difficult to achieve.

In this case, it is more sensible to do the predictions at the phrase level rather than further exploiting them at the sentence level. Therefore, we move our focus on phrase prediction in the next chapter and present a learning framework for phrase translations.

Chapter 5

The application of structured learning in natural language processing

Symbol	Notation
$\mathbf{f}$	the source sentence (string)
$\mathbf{e}$	the target sentence (string)
f_j	the j-th word in the source sentence
e_i	the i-th word in the target sentence
$\bar{\mathbf{f}}^I$	the source phrase sequence
$\bar{\mathbf{e}}^I$	the target phrase sequence
$\bar{f}_j$	the source phrase where $\bar{f}$ denotes the sequence of words $[f_{j_l}, \ldots, f_{j_r}]$ and j denotes that $\bar{f}_j$ is the j-th phrase in $\bar{\mathbf{f}}^I$
$\bar{e}_i$	the target phrase where $\bar{e}$ denotes the sequence of words $[e_{i_l}, \ldots, e_{i_r}]$ and i denotes that $\bar{e}_i$ is the i-th phrase in $\bar{\mathbf{e}}^I$
$\Omega_{\bar{f}}$	the output space containing all translations for the source phrase $\bar{f}$
$C_{\bar{f}}$	the number of translations in $\Omega_{\bar{f}}$
$\mathcal{S}_{\bar{f}}$	a set of training examples for the source phrase $\bar{f}$
$N_{\bar{f}}$	the number of phrase pairs in $\mathcal{S}_{\bar{f}}$
$(\bar{f}_j^n, \bar{e}_i^n)$	the n-th example in $\mathcal{S}_{\bar{f}}$ that is also abbreviated as $(\bar{f}^n, \bar{e}^n)$
$\phi(\bar{f}_j^n, \bar{e}_i^n)$	the feature vector of the phrase pair $(\bar{f}_j^n, \bar{e}_i^n)$
d	the dimension of linguistic feature space
$\triangle(c, \bar{e}^n)$	the "distance" between a pseudo candidate c and the correct translation $\bar{e}^n$

Table 5.1: Notations used in this chapter.

5.1 Introduction

As discussed in Section 3.4.3, structured learning is now established as a powerful paradigm of formulating and solving difficult tasks in artificial intelligence, and has been increasingly applied in the domains of natural language processing (NLP) and statistical machine translation (SMT) in recent years. In this chapter, we propose a modern structured learning methodology for two specific problems in NLP: *phrase translation* in machine translation and *part-of-speech* (POS) *tagging*. In both tasks we wish to devise a methodology that is specifically aimed at capturing structure in the output predictions.

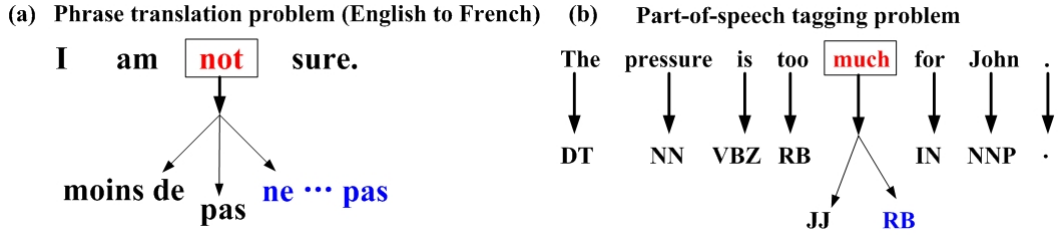


FIGURE 5.1: Examples of two structured prediction tasks: (a) phrase translation and (b) POS tagging. The target phrase (or POS tag) in blue is the correct output.

5.1.1 Phrase translation

Bilingual machine translation usually involves two types of activities: understanding a *source* text and formulating it with *target* language. As reviewed in Section 2.4, one state-of-the-art approach to machine translation is phrase-based SMT which involves the creation of a number of sub-models $\{h_m\}$, namely a *phrase translation probability* (PTP) model, a language model and a phrase reordering model. Then an SMT decoder is employed to solve the task where each source sentence $\mathbf{f}$ is segmented into a sequence of I phrases $\bar{\mathbf{f}}^I$ and translated into a target sequence $\bar{\mathbf{e}}^I$ such that $\bar{\mathbf{e}}^I = \arg\max_{\hat{\mathbf{e}}^I \in E} \left\{ \exp\left(\sum_m \lambda_m h_m(\hat{\mathbf{e}}^I, \bar{\mathbf{f}}^I) \right) \right\}$. This is equivalent to searching a Viterbi–best string path according to the decoding information provided by the sub-models.

These individual sub-models are designed specifically for capturing different information in the translation process. For example, the PTP model aims at modelling semantics and disambiguating phrases; while the goal of the language and the phrase reordering models is to predict syntactic structure in the target language domain and help generate fluent translations. Therefore, the design of these sub-models is critical for the success of an SMT system.

In this chapter, we focus on developing the *phrase translation probability* (PTP) model, whose main task is to predict the *target translation* from a finite candidate pool for a *source phrase*, on the basis of the linguistic environment (or context) in which the source phrase is embedded (see Figure 5.1 (a)). Although a better phrase translation

prediction can greatly improve the translation quality as well as ease the workload of the SMT decoder, the PTP model has changed little over the past few years.

As machine learning techniques become more pervasive in SMT, several discriminative methods (Vickrey et al., 2005; Carpuat and Wu, 2007; Giménez and Màrquez, 2007) have been applied for the PTP model. Compared with the traditional PTP model that utilises maximum likelihood estimation (MLE), their capability in exploring the sentence context brings in advantages that the pure statistical methods do not bring. Nevertheless, among these discriminative models the relationships between target translations are still not considered at all. Theoretically, the structure of the target translations can be learnt by state-of-the-art structured prediction technologies described in Section 3.4.3, but in practice the runtime usually makes it infeasible to apply these methods to even medium-sized data sets. Therefore, we use the idea of structured learning that allows more flexible margins between classes, but apply a perceptron-based algorithm in order to reduce the computational complexity. Our aim is to show that it is practical to apply this *max–margin structure* (MMS) model to the MT field and produce reasonable results.

5.1.2 Part–of–speech tagging

POS tagging can be viewed as the process of "translating" words in a text into a particular part of speech. Since the translation from words to POS tags are one by one without word reordering (see Figure 5.1 (b)), it can be regarded as a simplified form of machine translation that only consists of a phrase translation probability model.

Among recent top performing methods for automatic assignment of POS tagging, *hidden markov models* (Brants, 2000), *conditional random field models* (Lafferty et al., 2001) and *maximum entropy models* (Toutanova et al., 2003) are very popular. In these methods, a tag t_i assigned to a word f_i with the context feature function ϕ is connected via a conditional probability $p(\mathrm{t}_i|\phi(f_i))$ while different parametric forms are applied for modelling this probability. The models are then "generating" the POS tag sequence for a given sentence by maximising the sequence probability $\prod_{i=1}^{N} p(\mathrm{t}_i|\phi(f_i))$. Alternatively, one can view POS tagging as a multi–class classification problem, which predicts each word f_i's tag label t_i in accordance with its linguistic features: $\mathrm{t}_i = \arg\max_{\hat{\mathrm{t}}} \mathbf{w}^T \phi(f_i, \hat{\mathrm{t}})$. In this way, general classification techniques such as SVMs can be applied (Joachims, 1999).

In POS tagging there are several problematic cases that come from ambiguous words in speech. For example, the word "good" in the phrase "good strategy" is an adjective (JJ) while in "the common good" it is a noun (NN). Current methods such as (Toutanova et al., 2003) try to solve the problematic cases by exploring richer feature sets, such as grammatical features and lemmas. However, such feature extensions are usually expensive to collect in advance.

In contrast to the above, we aim at improving the performance by exploiting the potential structure in the output (tags) domain for ambiguous words (see Figure 5.2). The assumption is that for the POS tag candidates of an ambiguous word, some of them might be closer; and the learning agent taking this tag structure into account would achieve a better disambiguation performance. This is the same as the inspiration for the structured phrase prediction, which is discussed in Section 5.2.

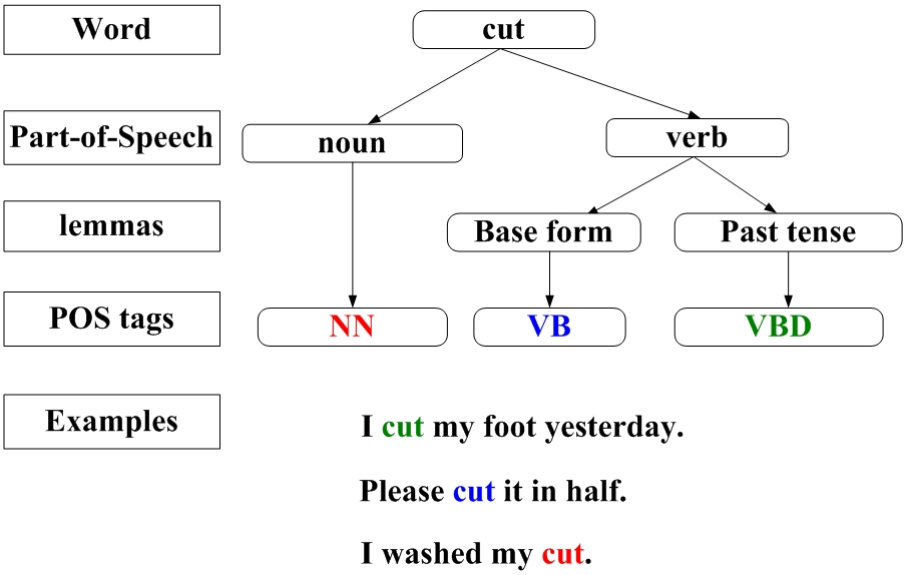


FIGURE 5.2: A part-of-speech tagging example for the word "cut". The latent connections between the tags are displayed in two levels: "part-of-speech" and "lemmas".

Under this assumption, we apply a structured word disambiguation technique to the POS tagging problem and utilise the *max–margin structure* (MMS) model as the learning agent. The model is then compared with two other agents – the *maximum entropy* (ME) framework and the *support vector machine* (SVM) technique[1] – to show its advantages. In this chapter we treat POS tagging as an MT task with a specific target language made of POS tags, where the goal is to show the effectiveness of the MMS model and pave the way for its application to the full MT tasks.

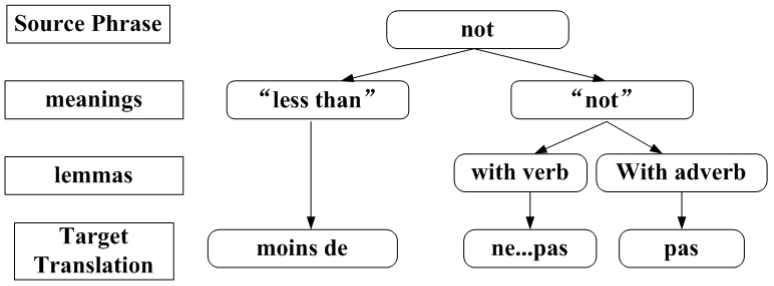
The remaining parts of this chapter are organised as follows: a general framework of the MMS model[2] is given in Section 5.2, which specifies the motivations of utilising the structured prediction and the learning algorithms. Then in Section 5.3, we demonstrate the procedure for feature extraction and the training scheme of the MMS model. Section 5.4 evaluates the performance of the MMS model on POS tagging and MT tasks. Finally, we draw conclusions and mention areas for future work in Section 5.5.

[1]The SVM technique mentioned in this chapter is the classic SVM optimisation Joachims (1999), which uses Sequential Minimal Optimisation as the SVM solver.

[2]Since the POS tagging is regarded as a special case of MT, in the system description our notations will follow the general MT notations as used in Chapter 2.

5.2 Phrase translation with structure exploitation

We define the source phrase as $\bar{f}_j^n$ with $\bar{f}$ denoting the phrase label, j denoting the phrase position of $\bar{f}_j$ in the phrase sequence $\bar{\mathbf{f}}^I$ and n denoting the n-th example. A similar notation $\bar{e}_i^n$ is used for the target phrases. Each unique source phrase $\bar{f}$ is assigned to a cluster (output space) $\Omega_{\bar{f}}$ that includes all possible target translations (candidates) and the number of candidates is denoted as $C_{\bar{f}}$. Figure 5.3 illustrates an English–to–French translation example, in which $\bar{f}$ = "not", $\Omega_{\bar{f}}$ = {"moins de", "ne...pas", "pas"} and $n = 1, \ldots, 3$. Whenever this can be done without loss of clarity, the source and the target phrases are also abbreviated as $\bar{f}^n$ and $\bar{e}^n$.



Example 1:
Source Sentence: Not five minutes ago.
Translation: Il y a moins de cinq minutes.
Example 2:
Source Sentence: I am not sure.
Translation: Je ne suis pas sûr.
Example 3:
Source Sentence: There is not even a pub.
Translation: Il n´y a même pas de bar.

FIGURE 5.3: An English–to–French translation example. The latent connections between target translations are displayed in two levels: "meaning" and "lemmas".

In our PTP model, we assign a separate sub-model for each unique source phrase $\bar{f}$. Assume a set of training instances $\mathcal{S}_{\bar{f}} = \{(\bar{f}^n, \bar{e}^n)\}_{n=1}^{N_{\bar{f}}}$ with the same source phrase $\bar{f}$, each of which consists of a structured feature vector $\phi(\bar{f}^n, \bar{e}^n) \in \mathbb{R}^{d \cdot C_{\bar{f}}}$ with d denoting the dimension of linguistic feature space. Then the goal is to learn a linear evaluation function

$$F := \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, \bar{e}^n) \rightarrow \mathbb{R} \tag{5.1}$$

that can "generate" an appropriate translation score for $(\bar{f}^n, \bar{e}^n)$.

Instead of applying MLE, we adopt a discriminative framework, the max–margin formulation of (Taskar et al., 2003), to find a mapping operator $\mathbf{w}_{\bar{f}} \in \mathbb{R}^{d \cdot C_{\bar{f}}}$ such that

$$\arg\max_{c \in \Omega_{\bar{f}}} \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c) \approx \bar{e}^n, \ \forall n.$$

This is equivalent to minimising a risk function $J(\mathbf{w}_{\bar{f}})$, which corresponds to the sum of all classification errors associated with the translation candidates

$$J(\mathbf{w}_{\bar{f}}) = \frac{1}{N_{\bar{f}}} \sum_{n=1}^{N_{\bar{f}}} \rho(\bar{f}^n, \bar{e}^n, \mathbf{w}_{\bar{f}}) + \frac{\lambda}{2} \|\mathbf{w}_{\bar{f}}\|^2 \tag{5.2}$$

where ρ is a specific loss function and $\lambda \geq 0$ is a regularisation parameter. The risk function (5.2) can generate a set of discriminative models with different loss functions $\rho(\bar{f}^n, \bar{e}^n, \mathbf{w}_{\bar{f}})$. For example, if we view phrase translation merely as a multi-class classification problem and let $\rho(\bar{f}^n, \bar{e}^n, \mathbf{w}_{\bar{f}}) = \max\{0, 1 - \xi(\bar{f}^n, \bar{e}^n) - \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, \bar{e}^n)\}$, then the model turns out to be a one-class SVM (Schölkopf et al., 2001).

As translations for the same source phrase tend to be interdependent, introducing flexible margins to separate different translation candidates sounds more reasonable. Consider Figure 5.3, if the phrase "not" is translated into "pas" instead of "ne...pas", intuitively the loss should be smaller than when it is translated into "moins de". In other words, the output (target translation) domain has an inherent structure (e.g. surface form and lemma) and the loss function should respect this. This intuition is inspired by the multi-category structure used in text categorisation, for which (Rousu et al., 2005) has shown that a loss function respecting this structure can achieve better classification performance . Hence, we model the phrase translation task as a *max-margin structure* (MMS) problem and apply a soft–margin loss on the structured labels:

$$\rho(\bar{f}^n, \bar{e}^n, \mathbf{w}_{\bar{f}}) = \max\{0, \max_{c \neq \bar{e}^n} [\triangle(c, \bar{e}^n) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c)] - \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, \bar{e}^n)\} \tag{5.3}$$

where $\triangle(c, \bar{e}^n)$ is applied to measure the "distance" between a pseudo candidate c and the correct translation $\bar{e}^n$. Theoretically, this loss requires that a pseudo candidate c which is "far away" from the true translation $\bar{e}^n$ must be classified with a large margin $\triangle(c, \bar{e}^n)$ while nearby candidates are allowed to be classified with a smaller margin.

This formulation is similar to the structured SVM discussed in Section 3.4.3, which solves the following optimisation problem

$$\begin{aligned} \min_{\mathbf{w}_{\bar{f}}, \xi} \quad & \tfrac{1}{2}\|\mathbf{w}_{\bar{f}}\|^2 + \tfrac{C}{N} \sum_{n=1}^{N} \xi_n \\ s.t. \quad & \forall n,\, c \in \{\Omega_{\bar{f}} \backslash \bar{e}^n\} : \\ & \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, \bar{e}^n) - \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c) \geq \triangle(c, \bar{e}^n) - \xi_n \\ & \forall n \quad \xi_n \geq 0 \end{aligned}$$

However, instead of loading $NC_{\bar{f}}$ constraints, MMS only consists of N constraints and hence speeds up the training procedure.

A variety of approaches have been suggested to evaluate the "distance" between strings, such as the "bag–of–words" method for string kernels (Shawe-Taylor and Cristianini,

2004). Ideally, the measure function $\triangle(c, \bar{e}^n)$ should respect all aspects of influence on candidate connections, which is hard to achieve in practice however. To avoid using complex syntax information such as syntactical trees, which involves additional NLP tasks; we use a generalisation of the hamming distance – Levenshtein distance, that measures the minimum number of modifications required to change one string into another. The algorithm can be found in (Gusfield, 1997) and the distance measure function is of the form

$$\triangle(c, \bar{e}^n) = \frac{LevDist(c, \bar{e}^n)}{\max\limits_{c' \in \Omega_{\bar{f}}} LevDist(c', \bar{e}^n)} \tag{5.4}$$

where $LevDist(c, \bar{e}^n)$ returns the Levenshtein distance between strings c and $\bar{e}^n$. This distance measurement satisfies $\triangle(c, \bar{e}^n) \in (0, 1]$ with $\triangle(c, \bar{e}^n) = 1$ indicating that c is the farthest from $\bar{e}^n$ among the candidates. Note other distances, such as those based on factored representations (Hofmann et al., 2003; Cai and Hofmann, 2004) could also be considered.

5.2.1 Perceptron–based structured learning (PSL)

As mentioned in Chapter 2, one practical criterion to evaluate the MT system is the translation speed. In order to facilitate the training process, we ignore the regularisation term in (5.2) (i.e. $\lambda = 0$) and propose using a perceptron–based structured learning (PSL) algorithm to tune the parameters $\mathbf{w}_{\bar{f}}$. The pseudo code of the algorithm is given in Table 5.2. Analogous to the standard Novikoff theorem, we provide an upper bound on the number of updates and a lower bound on the achievable margin for the PSL algorithm. Note that this algorithm is an extension of *perceptron-based structured learning* discussed in Section 3.4.3, where in the latter case the output structure is ignored (i.e. $\triangle(c, \bar{e}^n) = 1,\ \forall c$).

Note that the margin between the hyperplane $\mathbf{w}_{\bar{f}}^T \phi(\bar{f}, \bar{e}) = 0$ and the data cloud is computed by

$$\gamma(\mathbf{w}_{\bar{f}}, \mathcal{S}_{\bar{f}}, \phi) := \min_{(\bar{f}^n, \bar{e}^n) \in \mathcal{S}_{\bar{f}}} \frac{\langle \mathbf{w}_{\bar{f}}, \phi(\bar{f}^n, \bar{e}^n) - \phi(\bar{f}^n, c_n^*) \rangle}{\|\mathbf{w}_{\bar{f}}\|}$$

with c_n^* to be the maximizer of the $\max_{c \neq \bar{e}^n}$ operation in equation (5.3). Then we have:

Proposition 5.1. *Let $\mathcal{S}_{\bar{f}} = \{(\bar{f}^n, \bar{e}^n)\}_{n=1}^{N_{\bar{f}}}$ be a sample set independently and identically drawn from an unknown distribution and let $\phi(\bar{f}^n, \bar{e}^n)$ be a feature vector with $\|\phi(\bar{f}^n, \bar{e}^n)\| = 1$ for all n, and that the learning rate η is a fixed positive number in Table 5.2. Suppose there exists a mapping operator $\mathbf{w}_{\bar{f}}^*$ such that $\|\mathbf{w}_{\bar{f}}^*\| = R$ and $\gamma(\mathbf{w}_{\bar{f}}^*, \mathcal{S}, \phi) \geq \Gamma$, and the algorithm stops when the functional margin V in Table 5.2 is achieved for every data point.*

Input of the learner: The samples $\mathcal{S}_{\bar{f}} = \{(\bar{f}^n, \bar{e}^n)\}_{n=1}^{N_{\bar{f}}}$, learning rate η
Initialisation: $t = 0$; $\mathbf{w}_{\bar{f}}^t = \mathbf{0}$;
Repeat
  randomly sample $(\bar{f}_n, \bar{e}_n) \in \mathcal{S}_{\bar{f}}$ **do**
    read input: $\phi(\bar{f}_n, \bar{e}_n) \in \mathbb{R}^{d \cdot C_{\bar{f}}}$;
    $V = \max_{c \neq \bar{e}^n} \{\triangle(c, \bar{e}^n) + \langle \mathbf{w}_{\bar{f}}^t, \phi(\bar{f}^n, c)\rangle\}$
    $c^* = \arg\max_{c \neq \bar{e}^n} \{\triangle(c, \bar{e}^n) + \langle \mathbf{w}_{\bar{f}}^t, \phi(\bar{f}^n, c)\rangle\}$
    if $\langle \mathbf{w}_{\bar{f}}^t, \phi(\bar{f}^n, \bar{e}^n)\rangle < V$ **then**
      $\mathbf{w}_{\bar{f}}^{t+1} = \mathbf{w}_{\bar{f}}^t + \eta\big(\phi(\bar{f}^n, \bar{e}^n) - \phi(\bar{f}^n, c^*)\big)$
      $t = t + 1$
    end if
until converge
Output of the learner: $\mathbf{w}_{\bar{f}}^{t+1} \in \mathbb{R}^{d \cdot C_{\bar{f}}}$

TABLE 5.2: Pseudo-code of the *perceptron–based structured learning* (PSL) algorithm.

1. *Then the number of updates made by the PSL algorithm is bounded by*

$$t \leq \frac{2}{\Gamma^2}(1 + \frac{1}{\eta}). \tag{5.5}$$

2. *Then for the solution* $\mathbf{w}_{\bar{f}}^t$ *of the PSL algorithm we have*

$$\gamma(\mathbf{w}_{\bar{f}}^t, \mathcal{S}_{\bar{f}}, \phi) \geq \frac{\Gamma\xi}{2(\eta + 1)} \tag{5.6}$$

with $\xi = \min_k \triangle(c_k^*, \bar{e}^k)$ *indicating the minimal distance between a pseudo candidate and a correct translation across all examples.*

Proof. First after t updates, we can upper bound the norm of $\mathbf{w}_{\bar{f}}^t$ by

$$\begin{aligned} \|\mathbf{w}_{\bar{f}}^t\|^2 &= \|\mathbf{w}_{\bar{f}}^{t-1} + \eta\big(\phi(\bar{f}^t, \bar{e}^t) - \phi(\bar{f}^t, c_t^*)\big)\|^2 \\ &= \|\mathbf{w}_{\bar{f}}^{t-1}\|^2 + 2\eta\langle \mathbf{w}_{\bar{f}}^{t-1}, \phi(\bar{f}^t, \bar{e}^t) - \phi(\bar{f}^t, c_t^*)\rangle \\ &\quad +\eta^2\|\phi(\bar{f}^t, \bar{e}^t) - \phi(\bar{f}^t, c_t^*)\|^2 \\ &\leq \|\mathbf{w}_{\bar{f}}^{t-1}\|^2 + 2\eta\triangle(c_t^*, \bar{e}^t) + 2\eta^2 \\ &\leq 2t\eta(1 + \eta) \end{aligned} \tag{5.7}$$

Then we provide a reverse inequality for the inner product with $\mathbf{w}_{\bar{f}}^*$

$$\begin{aligned} \langle \mathbf{w}_{\bar{f}}^t, \mathbf{w}_{\bar{f}}^*\rangle &= \langle \mathbf{w}_{\bar{f}}^{t-1}, \mathbf{w}_{\bar{f}}^*\rangle + \eta\langle \mathbf{w}_{\bar{f}}^*, \phi(\bar{f}^t, \bar{e}^t) - \phi(\bar{f}^t, c_t^*)\rangle \\ &\geq \langle \mathbf{w}_{\bar{f}}^{t-1}, \mathbf{w}_{\bar{f}}^*\rangle + \eta\Gamma R \\ &\geq t\eta\Gamma R \end{aligned} \tag{5.8}$$

By combining the above two inequalities, we have

$$2t\eta(1+\eta)\|\mathbf{w}^*_{\bar{f}}\|^2 \geq \|\mathbf{w}^t_{\bar{f}}\|^2\|\mathbf{w}^*_{\bar{f}}\|^2 \geq \langle \mathbf{w}^t_{\bar{f}}, \mathbf{w}^*_{\bar{f}}\rangle^2 \geq (t\eta\Gamma R)^2 \tag{5.9}$$

which yields

$$t \leq \frac{2}{\Gamma^2}(1+\frac{1}{\eta}).$$

For the achievable margin, taking the bound (5.5) for t and substituting it into (5.7), we arrive at

$$\|\mathbf{w}^t_{\bar{f}}\| \leq \frac{2(\eta+1)}{\Gamma} \tag{5.10}$$

Then for the margin we have

$$\begin{array}{rcl} \gamma(\mathbf{w}^t_{\bar{f}}, \mathcal{S}_{\bar{f}}, \phi) & \geq & \min\limits_{(\bar{f}^k, \bar{e}^k)} \frac{\langle \mathbf{w}^t_{\bar{f}}, \phi(\bar{f}^k, \bar{e}^k) - \phi(\bar{f}^k, c^*_k))\rangle}{\|\mathbf{w}^t_{\bar{f}}\|} \\ & \geq & \frac{\min\limits_k \triangle(c^*_k, \bar{e}^k)}{\|\mathbf{w}^t_{\bar{f}}\|} \\ & \geq & \frac{\Gamma\xi}{2(\eta+1)} \end{array} \tag{5.11}$$

where $\xi = \min\limits_k \triangle(c^*_k, \bar{e}^k)$ is the minimal distance between a pseudo candidate and a correct translation across all examples. □

Table 5.2 shows that the computational complexity of the PSL algorithm is $O(N_{\bar{f}}C_{\bar{f}}d)$, while the complexity of a multi-class SVM is somewhere between $O(N^2_{\bar{f}}d + N_{\bar{f}}C_{\bar{f}})$ and $O(N^2_{\bar{f}}d + N^2_{\bar{f}}C_{\bar{f}})$ (Bishop, 2006). Since in practice the number of classes $C_{\bar{f}}$ is much smaller than the number of examples $N_{\bar{f}}$, this makes PSL substantially faster than the multi–class SVM used in (Giménez and Màrquez, 2007) and most of the structured learning approaches discussed in Section 3.4.3. This time efficiency is also verified by the POS tagging experiment results shown in Section 5.4.

Although PSL does not adopt a regularisation term, implying a potential risk of over-fitting, the *early stopping strategy*[3], which involves a careful design of the maximum number of iteration, can usually help to avoid this problem. If one wished to add regularisation to the model to further guard against over-fitting, one could apply methods such as ALMA (Gentile, 2001) or NORMA (Kivinen et al., 2004). However, the requirement of normalising $\mathbf{w}_{\bar{f}}$ at each step makes the implementation intractable for a large learning problem. As an alternative, the risk function (5.2) can be reformulated as a min–max optimisation problem as shown in (Taskar et al., 2006), which can be solved by a benchmark-based learning algorithm.

[3]The strategy selects the maximum number of iterations and the learning rate η by cross-validating on a validation set. In our experiments, this was done on the POS tagging data and the (max-iteration, learning rate) with the best performance was chosen for both POS tagging and MT experiments.

5.2.2 Benchmark–based training – extra–gradient algorithm

To consider adding a regularisation term, we upper bound the norm of $\mathbf{w}_{\bar{f}}$ by $\|\mathbf{w}_{\bar{f}}\| \leq R$. Then minimising (5.2) with respect to $\mathbf{w}_{\bar{f}}$ is equivalent to solving the following optimisation problem

$$\min_{\|\mathbf{w}_{\bar{f}}\| \leq R} L(\mathbf{w}_{\bar{f}}, \mathcal{S}_{\bar{f}}, \phi) \tag{5.12}$$

where the cumulative loss $L(\mathbf{w}_{\bar{f}}, \mathcal{S}_{\bar{f}}, \phi) = \sum_n \rho(\bar{f}^n, \bar{e}^n, \mathbf{w}_{\bar{f}})$.

Generally, we can express the sub maximisation problem $\max_{c \neq \bar{e}^n}[\triangle(c, \bar{e}^n) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c)] - \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, \bar{e}^n)$ as a *Linear Programming* problem

$$\begin{array}{ll} \max\limits_{\mathbf{z}_n} & \sum\limits_{c \in \Omega_{\bar{f}}} z_n^c [\triangle(c, \bar{e}^n) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c)] \\ s.t. & \sum\limits_c z_n^c = 0 \\ & z_n^{\bar{e}^n} = -1 \\ & z_n^c \geq 0,\ c \in \{\Omega_{\bar{f}} \backslash \bar{e}^n\} \end{array} \tag{5.13}$$

Let $\mathcal{Z}_n$ denote the closed set of $\mathbf{z}_n$, $\mathbf{z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ and $\mathcal{Z} = \mathcal{Z}_1 \times \ldots \times \mathcal{Z}_N$, then substituting (5.13) into (5.12) yields a natural saddle–point form

$$\min_{\|\mathbf{w}_{\bar{f}}\| \leq R} \max_{\mathbf{z} \in \mathcal{Z}} L(\mathbf{w}_{\bar{f}}, \mathbf{z}) \tag{5.14}$$

with

$$L(\mathbf{w}_{\bar{f}}, \mathbf{z}) = \sum_n \max \Big\{ 0, \sum_c z_n^c \big(\triangle(c, \bar{e}^n) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^n, c) \big) \Big\}$$

A well–known solution for the saddle–point optimisation is provided by the extra-gradient method, which consists of two simple steps in an iteration:

$$\text{(Prediction)} \quad \begin{cases} \bar{\mathbf{w}}_{\bar{f}}^{t+1} = \mathscr{P}_R(\mathbf{w}_{\bar{f}}^t - \eta \nabla_{\mathbf{w}_{\bar{f}}} L(\mathbf{w}_{\bar{f}}^t, \mathbf{z}^t)) \\ \bar{\mathbf{z}}_n^{t+1} = \mathscr{P}_{\mathcal{Z}_n}(\mathbf{z}_n^t + \eta \nabla_{\mathbf{z}_n} L(\mathbf{w}_{\bar{f}}^t, \mathbf{z}^t)) \end{cases} \tag{5.15}$$

$$\text{(Correction)} \quad \begin{cases} \mathbf{w}_{\bar{f}}^{t+1} = \mathscr{P}_R(\mathbf{w}_{\bar{f}}^t - \eta \nabla_{\mathbf{w}_{\bar{f}}} L(\bar{\mathbf{w}}_{\bar{f}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \\ \mathbf{z}_n^{t+1} = \mathscr{P}_{\mathcal{Z}_n}(\mathbf{z}_n^t + \eta \nabla_{\mathbf{z}_n} L(\bar{\mathbf{w}}_{\bar{f}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \end{cases} \tag{5.16}$$

In (5.15) and (5.16), $\mathscr{P}_{\mathcal{H}}(\mathbf{x})$ denotes the projection of vector $\mathbf{x}$ on space $\mathcal{H}$ and the partial derivatives of the cumulative loss are given by

$$\nabla_{\mathbf{w}_{\bar{f}}} L(\mathbf{w}_{\bar{f}}, \mathbf{z}) = \sum_{k \in \mathcal{K}} \sum_{c \in \Omega_{\bar{f}}} z_k^c \phi(\bar{f}^k, c) \tag{5.17}$$

$$\begin{array}{ll} \nabla_{z_k^c} L(\mathbf{w}_{\bar{f}}, \mathbf{z}) & = \triangle(c, \bar{e}^k) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^k, c) \\ w.r.t & k \in \mathcal{K} \text{ and } c \in \{\Omega_{\bar{f}} \backslash \bar{e}^n\} \end{array} \tag{5.18}$$

where $\mathcal{K} = \{k : \sum_c z_k^c[\triangle(c, \bar{e}_k) + \mathbf{w}_{\bar{f}}^T \phi(\bar{f}^k, c)] > 0\}$ is the set of *support vectors*.

Finally, by defining the projection of $\mathbf{w}$ onto a ball $\|\mathbf{w}_{\bar{f}}\| \leq R$ as $\mathscr{P}_R(\mathbf{w}_{\bar{f}}) = R\mathbf{w}_{\bar{f}} / \max(R, \|\mathbf{w}_{\bar{f}}\|)$ and the projection of $\mathbf{z}_n$ onto $\mathcal{Z}_n$ as

$$\mathscr{P}_{\mathcal{Z}_n}(z_n^c) = \begin{cases} z_n^c / (\|\mathbf{z}_n\| - 1) & \text{if } c \neq \bar{e}^n \\ -1 & \text{if } c = \bar{e}^n \end{cases},$$

the iterative solution can be obtained (see Table 5.3). Under mild conditions, this method is guaranteed to converge linearly to a solution of $\mathbf{w}_{\bar{f}}^*$ and $\mathbf{z}^*$ (Taskar et al., 2006).

Input of the learner: The samples $\mathcal{S}_{\bar{f}} = \left\{(\bar{f}^n, \bar{e}^n)\right\}_{n=1}^{N_{\bar{f}}}$
Initialisation: Initialise a starting value $\mathbf{w}_{\bar{f}}^0$ and $\mathbf{z}^0$
Repeat
a) Prediction step for $\mathbf{w}_{\bar{f}}$ and $\mathbf{z}$ using the update rule

$$\begin{cases} \bar{\mathbf{w}}_{\bar{f}}^{t+1} = \mathscr{P}_R(\mathbf{w}_{\bar{f}}^t - \eta \nabla_{\mathbf{w}_{\bar{f}}} \mathcal{L}(\mathbf{w}_{\bar{f}}^t, \mathbf{z}^t)) \\ \bar{\mathbf{z}}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\mathbf{w}_{\bar{f}}^t, \mathbf{z}^t)) \end{cases}$$

b) Correction step for $\mathbf{w}_{\bar{f}}$ and $\mathbf{z}$ using the update rule

$$\begin{cases} \mathbf{w}_{\bar{f}}^{t+1} = \mathscr{P}_R(\mathbf{w}_{\bar{f}}^t - \eta \nabla_{\mathbf{w}_{\bar{f}}} \mathcal{L}(\bar{\mathbf{w}}_{\bar{f}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \\ \mathbf{z}^{t+1} = \mathscr{P}_{\mathcal{Z}}(\mathbf{z}^t + \eta \nabla_{\mathbf{z}} \mathcal{L}(\bar{\mathbf{w}}_{\bar{f}}^{t+1}, \bar{\mathbf{z}}^{t+1})) \end{cases}$$

until converge
Output of the learner: $\mathbf{w}_{\bar{f}}^{t+1} \in \mathbb{R}^{d \cdot C_{\bar{f}}}$

TABLE 5.3: Pseudo-code of the benchmark–based *extra–gradient* algorithm.

5.3 Training procedure

In this section, we describe two key steps for the proposed method: feature extraction and model training.

5.3.1 Feature extraction

Following (Vickrey et al., 2005), we consider different kinds of information extracted from the phrase environment (see Figure 5.4). The types of features used are depicted in Table 5.4.

To specify the difference with respect to each source environment position d_z, we express the features as

$$\phi_u(s_p^{|u|}) = \delta(s_p^{|u|}, u),$$

with the indicator function $\delta(\cdot, \cdot)$, $p = \{j_l - d_l, \ldots, j_l, j_r, \ldots, j_r + d_r\}$ and string $s_p^{|u|} = [f_p, \ldots, f_{p+|u|}]$ with $|u|$ denoting the length of u. In this way, the linguistic features are

Types	Feature Extraction
Context	Word n–grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$
Syntax	Part-of-speech tagn-grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$

TABLE 5.4: Features extracted from the phrase environment. n-gram indicates a word sequence of length n.

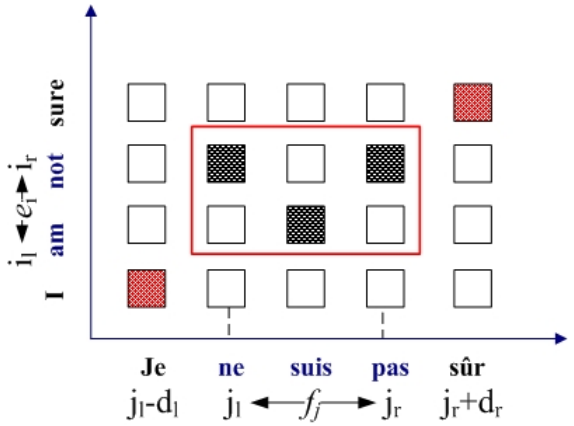


FIGURE 5.4: Illustration of the phrase pair ("ne suis pas","am not") (the word alignments are in black boxes). The linguistic features are extracted from a window environment (red shadow boxes) around the source phrase.

distinguished by both the content u and the start position p. For example, in Figure 5.3 the word "not" in example 2 has the following context features

$$\{\delta(s_0^1, \text{“I”}), \delta(s_1^1, \text{“am”}), \delta(s_3^1, \text{“sure”}), \delta(s_0^2, \text{“I am”})\}.$$

As required by the PSL algorithm, we then *normalise* the feature vector $\bar{\phi}_t = \frac{\phi_t}{\|\phi\|}$.

5.3.2 Model training

To form the training sample pool, all consistent phrase pairs $\{(\bar{f}_n, \bar{e}_n)\}_{n=1}^{N_{\bar{f}}}$ with the corresponding features are derived from the training sentences using the phrase pair extraction procedure described in Section 2.4.2[4]. Then the instances having the same source phrase $\bar{f}$ are considered to be from the same cluster (see Figure 5.3 for example) and a mapping operator $\mathbf{w}_{\bar{f}}$ is tuned by the cluster samples only. When decoding, given

[4]Since there is no word alignment problem in POS tagging experiments, the word–to–tag samples with the linguistic features are derived directly from the training corpus.

a source phrase $\bar{f}_j$, we find the corresponding cluster model and predict the confidence–rated possibility for each candidate translation. For MT experiments, the confidence–rated values are then transformed to probabilities using the softmax function (Bishop, 2006)

$$p(\bar{f}_j, \bar{e}_i) = \frac{\exp\{\mathbf{w}_{\bar{f}}^T \phi(\bar{f}_j, \bar{e}_i)\}}{\sum_{c \in \Omega_{\bar{f}}} \exp\{\mathbf{w}_{\bar{f}}^T \phi(\bar{f}_j, c)\}}. \tag{5.19}$$

5.4 Experiments

5.4.1 Part–of–speech (POS) tagging

In this chapter, we regard POS tagging as a special case of machine translation. The motivation of this experiment is to introduce the MMS model for capturing relationships between the tags for ambiguous words, and we expect it to improve the performance of problematic cases described in (Santorini, 1990). In contrast to MT, the "distances" between POS tags for the same word (e.g. Figure 5.2) are unknown in advance and can not be measured by the Levenshtein distance. To avoid additional work in designing a tag domain structure, the distance matrix $\triangle(\mathrm{t}_i, \mathrm{t}_j)$ used is pre-defined heuristically, according to the problematic cases[5] described in (Santorini, 1990). In general, the harder the problematic case is, the larger the distance will be. For two tags t_i and t_j, the distance is pre-defined as

$$\triangle(\mathrm{t}_i, \mathrm{t}_j) = \begin{cases} 0 & \text{if } \mathrm{t}_i = \mathrm{t}_j \\ 1 & \text{if } \mathrm{t}_i, \mathrm{t}_j \text{ have no relationships} \\ \text{see Table 5.5} & \text{otherwise} \end{cases}$$

$\triangle(\text{“}JJ\text{”}, \text{“}VBD\text{”}) = 1.1$	$\triangle(\text{“}JJ\text{”}, \text{“}VBN\text{”}) = 1.1$	$\triangle(\text{“}NN\text{”}, \text{“}VB\text{”}) = 1.05$
$\triangle(\text{“}NN\text{”}, \text{“}VBD\text{”}) = 1.1$	$\triangle(\text{“}NN\text{”}, \text{“}VBG\text{”}) = 1.1$	$\triangle(\text{“}NN\text{”}, \text{“}VBN\text{”}) = 1.1$
$\triangle(\text{“}NN\text{”}, \text{“}VBP\text{”}) = 1.05$	$\triangle(\text{“}NNP\text{”}, \text{“}NNPS\text{”}) = 0.95$	$\triangle(\text{“}NNS\text{”}, \text{“}NNPS\text{”}) = 1.2$
$\triangle(\text{“}VB\text{”}, \text{“}VBG\text{”}) = 0.9$	$\triangle(\text{“}VB\text{”}, \text{“}VBG\text{”}) = 0.9$	$\triangle(\text{“}VB\text{”}, \text{“}VBZ\text{”}) = 0.95$
$\triangle(\text{“}VB\text{”}, \text{“}VBD\text{”}) = 0.85$	$\triangle(\text{“}VBD\text{”}, \text{“}VBN\text{”}) = 1.1$	$\triangle(\text{“}VBD\text{”}, \text{“}VBP\text{”}) = 1.05$
$\triangle(\text{“}VBD\text{”}, \text{“}VBG\text{”}) = 1.05$	$\triangle(\text{“}VBG\text{”}, \text{“}VBN\text{”}) = 1.03$	$\triangle(\text{“}VBN\text{”}, \text{“}VBP\text{”}) = 1.05$
$\triangle(\text{“}VBN\text{”}, \text{“}VBZ\text{”}) = 1.15$	$\triangle(\text{“}VBP\text{”}, \text{“}VBZ\text{”}) = 1.05$	

TABLE 5.5: The "distance" between POS tags for the problematic cases.

[5]The problematic cases in POS tagging are a number of difficult tagging decisions, including dealing with parts of speech that are easily confused and tagging specific words and collocations that should be assigned tags different from the usual ones. The readers are referred to (Santorini, 1990) for a detailed descriptions of all problematic cases met.

Then the MMS model was trained and tested on two corpora: the CoNLL2004[6] and the CoNLL2009 data sets[7]. The former is the POS tagged Wall Street Journal section of the Penn Treebank, where sections 15–18 were used for training and section 20 as a test set. The latter is a larger POS tagged set which matches sections 2–21 and 24 of the Penn Treebank, where 40,000 training sentences and 613 test sentences were sampled from the corpus and the experiments were repeated three times to access the variance. The sizes of both data sets are shown in Table 5.6.

To compare the performance, results derived from two other systems are also displayed. One is the Stanford POS Tagger (Toutanova et al., 2003) that utilises a maximum entropy framework; the other is a multi–class SVM model which is trained by SVM–Multiclass (Joachims, 1999). The performance is measured by the word error rate (WER) as well as several class–specific F1 scores for the most problematic cases.

	CoNLL2004 data set		**CoNLL2009 data set**	
	Tokens	**Unknown Tokens**	**Tokens**	**Unknown Tokens**
Training	211,727	0	976,567	0
Test	47,377	3,092 (6.5%)	14,964	261(1.7%)

TABLE 5.6: Data sizes of POS tagging experiments.

Feature	**Description**
Capital	the word contains capital character(s)
number	the word contains number(s)
hyphen	the word contains hyphen symbol
"-ed"	the word ends with "ed"
"-ing"	the word ends with "ing"
"-s"	the word ends with "s"

TABLE 5.7: Word specific features for POS tagging.

The feature set for the Stanford system is described in (Toutanova et al., 2003) and a beam search decoder was applied to generate the predicted tag sequence. Alternatively, the MMS and the multi–class SVM models used the context features in Table 5.4, as well as the word specific features demonstrated in Table 5.7. Observing that POS features might help the prediction, we also made use of the syntactic features (see Table 5.4) by applying two–stage prediction. That is, first predicting the POS tags using the context features only, then predicting the POS tags again by incorporating the POS features predicted in the first stage. Since unknown words are unable to be assigned to certain word clusters, they were assigned to certain environment clusters instead. That is, for a sample with an unknown word f_j, it was assigned to a cluster with samples having

[6]Data supplied by Conference on Natural Language Learning (CoNLL) 2004 shared tasks and can be downloaded at `http://www.lsi.upc.edu/~srlconll/soft.html`.

[7]Data supplied by Conference on Natural Language Learning (CoNLL) 2009 shared tasks.

the same word environment $\{f_{j-1}, *, f_{j+1}\}$. For those unknown words which can't be assigned to any cluster, we simply regarded them as un-predictable.

The results are shown in Table 5.8. The WER figure for the MMS model is the lowest. For the CoNLL2004 (small) data set, it achieves a relative improvement of 6.1% over the Stanford system on the known words and 20.4% on the unknown words. For the CoNLL2009 (large) data set, the relative improvements increase to 8.6% on the known words and a particular 41.4% on the unknown words. Similar improvements are observed over the multi-class SVM, showing the outstanding ability of MMS in exploiting linguistic features. Table 5.9 depicts the class–specific F1 scores for different POS tags that have the most confusing cases. In many cases, the MMS model performs better than the multi–class SVM.

Table 5.8 also displays the runtime of the three models[8]. Compared with the other two learning agent, PSL is substantially faster. Figure 5.5 further depicts the runtime for training examples from $\{5k, 10k, 20k, 30k, 40k\}$ sentences using MMS and SVM respectively, demonstrating a linear time increase with MMS where in contrast a quadratic increase with SVM. This shows the time efficiency of the MMS model and suggests that it is more applicable to larger learning problems (e.g. MT problems).

	CoNLL2004 data set			CoNLL2009 data set		
Model	K–WER.	UN–WER.	Training	K–WER.	UN–WER.	Training
Stanford	6.07	34.74	1.71 hours	2.90 ± 0.15	31.6 ± 4.8	2.02 hours
SVM	5.98	27.86	2.19 hours	3.54 ± 0.15	23.8 ± 2.2	2.70 hours
SVM + POS	5.73	29.06	2.29 hours	2.80 ± 0.14	24.7 ± 1.5	2.80 hours
MMS	5.92	**27.65**	0.56 hour	3.2 ± 0.10	**18.6** ± 1.4	0.61 hours
MMS + POS	**5.70**	29.48	0.70 hour	**2.65** ± 0.10	23.0 ± 2.6	1.04 hours

TABLE 5.8: Test word error rate (WER) [%] for known words (K–WER.) and unknown words (UN–WER.) of the three systems. If not specified the models use the context and word specific features only; "POS" denotes using the predicted POS features as well. Bold numbers indicate the best results.

Tag	**F1 score**	**Tag**	**F1 score**	**Tag**	**F1 score**
IN	**98.2**% / 98.1%	JJ	**92.4**% / 92.3%	VBD	**80.6**% / 80.5%
NN	93.7% / 93.7%	NNP	96.9% / 96.9%	VBN	**69.9**% / 69.2%
NNPS	**52.4**% / 51.2%	RB	90.9% / 90.9%	VBP	77.0% / **77.2**%
RP	76.7% / **77.0**%	VB	75.4% / **75.6**%	VBZ	**86.5**% / 86.4%

TABLE 5.9: F1 scores for the most confusing POS classes, using "MMS + POS" (left) and "SVM + POS" (right). Bold numbers indicate better results.

[8]The Stanford system is coded in Java, the MMS model is coded in Python and SVM-multiclass is coded in C++.

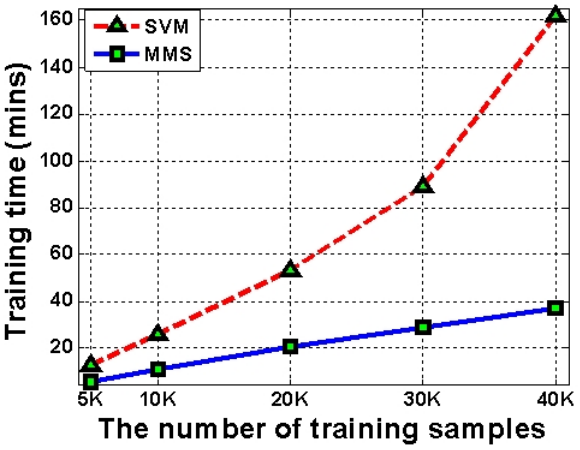


FIGURE 5.5: The runtime for training examples from $\{5k, 10k, 20k, 30k, 40k\}$ sentences using MMS (coded in Python) and SVM (coded in C++) respectively.

Overall, most of our observations of MMS are desirable properties for the design of a PTP model, particularly the PSL algorithm provides a promising learning agent that has a good performance on both classification accuracy and time efficiency.

5.4.2 Phrase translation in an MT system

In this experiment, we use MMS for two complex MT tasks: French-to-English and English-to-French translation using the *EuroParl* corpus[9]. Sentences of lengths between 1 and 100 words from the corpus were extracted where the ratio of source/target lengths was no more than 5 : 1. The training sets were taken between $\{50k, 100k\}$ sentences while the test set was fixed at $1k$ sentences. To compare the performance, two SMT systems – Pharaoh (Koehn, 2004) and MOSES (Koehn et al., 2005) – whose PTP models use maximum likelihood estimation (MLE), were taken as the baseline systems. To keep the comparison fair, our MT system just replaces their PTP models with our MMS predictor while sharing all other models (i.e. language model, phrase reordering model[10] and beam search decoder).

For parameter tuning, minimum-error-rating training (MERT) (Och, 2003) was applied. Experiments were repeated five times to assess variance and the performance was evaluated by four standard MT measurements, namely word error rate (WER), BLEU, NIST and METEOR[11].

We first demonstrate on Table 5.10 the phrase classification results on the $50k$–sentence tasks, using MLE, SVM and MMS respectively. In addition, Figure 5.6 compares MMS

[9]The corpus can be downloaded at `http://www.statmt.org/europarl/`.

[10]For MOSES, we only use the word distance–based reordering model to reduce the effect of the reordering model.

[11]The readers are referred to section 2.6 for details.

phrase classification	**MLE**	**SVM**	**MMS**
FR–to–EN	$64.8\% \pm 0.4\%$	$65.1\% \pm 0.4\%$	$\mathbf{65.6}\% \pm 0.1\%$
EN–to–FR	$52.8\% \pm 0.6\%$	$56.2\% \pm 0.3\%$	$\mathbf{56.8}\% \pm 0.3\%$

TABLE 5.10: The classification precisions on the $50k$–sentence tasks. Bold numbers refer to the best results.

with MLE on the basis of the overall accuracy of each cluster, suggesting when MMS works better and where it is better to apply:

- When given enough training samples (the black lines in Figure 5.6), MMS is usually better than MLE. This verifies the advantage of the discriminative model and suggests using MMS when the training samples reach a reasonable size.
- The vocabulary in French is larger than that in English, which causes more polysemies when translating English into French (represented by the increasing numbers of large circles in Figure 5.6(b)). The causes can be varied: the stylistic difference between English and French (e.g. English prefers the simple words from its Germanic wordstock where in contrast French uses learned words[12]); the different use of prepositions in French due to the grammatical gender (e.g. the English word "the" can be translated into "la", "le" and "les" in French, based on the noun it modifies); and the matter of French being more inflected than English. These situations make MMS a better choice, where the structured learning idea is beneficial. In contrast, when there is little ambiguity information in the target domain (i.e. less polysemies when translated French into English), MMS cannot benefit so much from the distance matrix and hence the positive effect of structured learning is not so high (represented by the smaller improvement on the French-to-English phrase classification task).

Table 5.11 depicts the translation results, where we observed consistent improvements in all evaluations. In addition, the improvements of MMS over the baselines on English-to-French MT tasks are usually better than those on French-to-English MT tasks, which is consistent with what has been observed on the phrase classification experiments. In particular, the WER and NIST scores concern more about the contents (rare n-grams), an improvement in both indicates that the MMS model is better in picking up correct words and phrases, which demonstrates the benefit of phrase disambiguation given by the MMS model.

[12]The "learned words" include a multitude of words which are comparatively seldom used in ordinary conversation. Their meanings are known to every educated person but there is little occasion to employ them at home or in the market-place (Greenough and Kittredge, 1914). For example, the word "eye" is a popular word in the ordinary conversation, while "ocular" is a learned word. The readers are referred to Chapter III in (Greenough and Kittredge, 1914) for the concept of "learned words"; the stylistic difference in using these words between English and French can be found in Section 2.1 to Section 2.2 in (Vinay and Darbelnet, 1995).

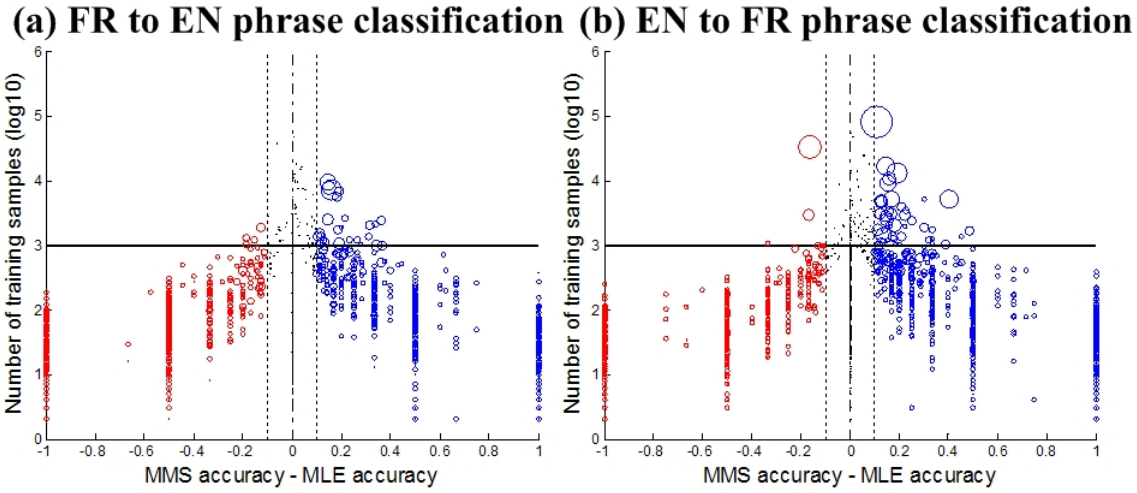


FIGURE 5.6: Scatter-plots comparing the cluster accuracies of the MMS model with the MLE model on $50k$-sentence French-to-English task (a) and English-to-French task (b). A cluster $\mathcal{S}_{\bar{f}}$ contains all phrase pairs with a unique source phrase $\bar{f}$. Those clusters for which the performance difference (x-axes) is greater than 0.1 are shown as circles, the areas of which are proportional to the number of target translations in them. The y-axes show the number of training samples (in log 10 form) for each cluster.

5.5 Conclusion and future work

In this chapter, we applied the max-margin structure (MMS) model for two related NLP tasks: *POS tagging* and *phrase translation* in machine translation. We have shown that when using certain distance measures between output classes (e.g. tags or target translations), the MMS model showed improved performance for both tasks. Furthermore the PSL algorithm is faster than SVM without decreasing the performance in practice, making this model more applicable to large scale learning problems (e.g. MT problems).

For future work, we will further develop our model for MT problems. We will refine the learning framework of MMS by carefully designing or automatically learning the distance matrix $\triangle(c, \bar{e}^n)$, aiming at capturing more complex structures in the target domain. Furthermore, a novel perceptron-based learning agent (Shalev-Shwartz et al., 2007) that utilises a stochastic gradient descent update claimed to have faster convergence rate. This motivates our further investigation of speed improvement in the PSL algorithm by incorporating a similar update strategy to that used in (Shalev-Shwartz et al., 2007).

From the analysis of classification performance, we observed that the MMS model did not perform equally well on all source phrase clusters, some results were even worse than MLE predictions. In this case, a hybrid PTP predictor that makes use of more than one ML technologies might be helpful in improving the classification performance, which is also of our interest.

Tasks	System	MT evaluations			
		BLEU [%]	WER [%]	NIST	METEOR [%]
FR–EN	MOSES	26.0 ± 0.2	39.2 ± 0.4	6.62 ± 0.06	48.7 ± 0.3
$50k$	MMS	$\mathbf{27.1} \pm 1.6$	$\mathbf{38.6} \pm 0.7$	$\mathbf{6.74} \pm 0.12$	$\mathbf{49.4} \pm 0.7$
EN–FR	MOSES	25.2 ± 0.3	43.0 ± 0.3	6.43 ± 0.04	47.8 ± 0.2
$50k$	MMS	$\mathbf{27.1} \pm 0.4$	$\mathbf{42.4} \pm 0.3$	$\mathbf{6.58} \pm 0.03$	$\mathbf{48.6} \pm 0.2$
FR–EN	Pharaoh	26.5 ± 0.4	39.0 ± 0.3	6.69 ± 0.04	50.8 ± 0.7
$100k$	MMS	$\mathbf{27.1} \pm 0.4$	$\mathbf{38.2} \pm 0.3$	$\mathbf{6.80} \pm 0.04$	$\mathbf{51.1} \pm 0.7$
EN–FR	Pharaoh	25.1 ± 0.4	42.6 ± 0.5	6.49 ± 0.06	47.9 ± 0.2
$100k$	MMS	$\mathbf{26.0} \pm 0.5$	$\mathbf{41.4} \pm 0.5$	$\mathbf{6.65} \pm 0.06$	$\mathbf{48.6} \pm 0.3$

P-value of T-test				
Task	MT Evaluations			
	BLEU	WER	NIST	METEOR
FR–EN ($50k$)	$8.10e-3$	$6.04e-4$	$1.26e-4$	$2.00e-3$
EN–FR ($50k$)	$1.30e-3$	$1.60e-2$	$7.40e-3$	$2.82e-4$
FR–EN ($100k$)	$8.10e-3$	$6.04e-4$	$1.26e-4$	$2.00e-3$
EN–FR ($100k$)	$1.30e-3$	$1.60e-2$	$7.40e-3$	$2.82e-4$

TABLE 5.11: Evaluations for MT experiments. Bold numbers refer to the best results. P-values of T-test for statistical significance in the differences between MMS and other systems are shown in the lower table.

In addition, we will focus on the integration between the MMS model and other MT models (e.g. language model, phrase reordering model), as performance could be improved if the influence of these models are more effectively balanced in an end-to-end MT system.

Finally we will try the MMS model on larger corpora (e.g. the whole EuroParl corpus), with the purpose of verifying its ability in scaling up to large data collections.

Chapter 6

Exploitation of ML techniques in modelling phrase movements for MT

Symbol	**Notation**
$\mathbf{f}$	the source sentence (string)
$\mathbf{e}$	the target sentence (string)
f_j	the j-th word in the source sentence
e_i	the i-th word in the target sentence
$\bar{\mathbf{f}}^I$	the source phrase sequence
$\bar{\mathbf{e}}^I$	the target phrase sequence
$\bar{f}_j$	the source phrase where $\bar{f}$ denotes the sequence of words $[f_{j_l}, \ldots, f_{j_r}]$ and j denotes that $\bar{f}_j$ is the j-th phrase in $\bar{\mathbf{f}}^I$
$\bar{e}_i$	the target phrase where $\bar{e}$ denotes the sequence of words $[e_{i_l}, \ldots, e_{i_r}]$ and i denotes that $\bar{e}_i$ is the i-th phrase in $\bar{\mathbf{e}}^I$
Υ	the set of the phrase pairs $(\bar{f}_j, \bar{e}_i) \in \Upsilon$
N	the number of examples in Υ
$(\bar{f}_j^n, \bar{e}_i^n)$	the n-th example in Υ that is also abbreviated as $(\bar{f}^n, \bar{e}^n)$
$\phi(\bar{f}_j, \bar{e}_i)$	the feature vector of the phrase pair $(\bar{f}_j, \bar{e}_i)$
d	the phrase reordering distance
o	the phrase reordering orientation
$\mathcal{O}$	the set of phrase reordering orientations $o \in \mathcal{O}$
$C_{\mathcal{O}}$	the number of phrase reordering orientations in $\mathcal{O}$
φ	embedding function to map the orientation set to an output space $\varphi : \mathcal{O} \rightarrow \mathbb{R}$
$\mathbf{w}_o$	weight vector measuring features' contribution to orientation o
$\{\mathbf{w}_o\}_{o \in \mathcal{O}}$	the set of weight vectors for the phrase reordering model
$K^{\phi}(n, m)$	the input kernel for the phrase pairs $(\bar{f}_j^n, \bar{e}_i^n)$ and $(\bar{f}_{j'}^m, \bar{e}_{i'}^m)$ $K^{\phi}(n, m) = \langle \phi(\bar{f}_j^n, \bar{e}_i^n), \phi(\bar{f}_{j'}^m, \bar{e}_{i'}^m) \rangle$
$K^{\varphi}(n, m)$	the orientation kernel for the phrase pairs $(\bar{f}_j^n, \bar{e}_i^n)$ and $(\bar{f}_{j'}^m, \bar{e}_{i'}^m)$ $K^{\varphi}(n, m) = \varphi(o_n, o_m)$
dim	the dimension of

TABLE 6.1: Notations used in this chapter.

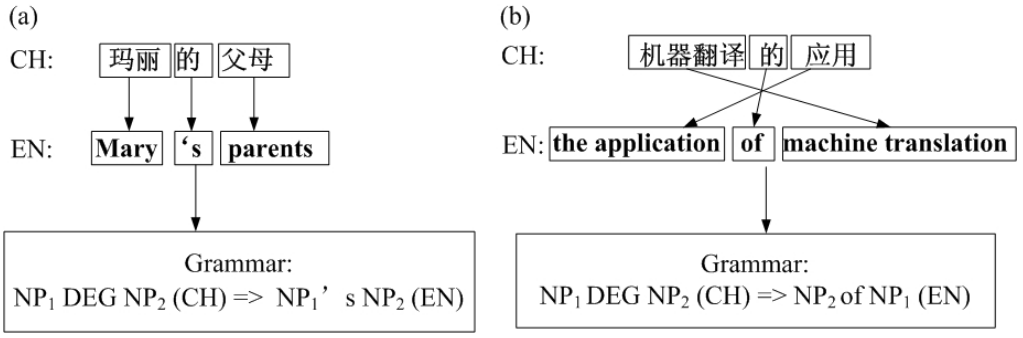


FIGURE 6.1: Example: the distance phrase reordering in Chinese-English bilingual translation.

6.1 Introduction

In this chapter, we focus on developing another crucial component in statistical machine translation (SMT) – the *phrase reordering model*, which is labeled as $p_d(\mathbf{f}, \hat{\mathbf{e}})$ in the SMT system (2.4). Word or phrase reordering is a common problem in bilingual translations arising from different grammatical structures. For example, the Chinese "NP_1 DEG NP_2" sequence is analogous to the English possessive structure of "NP_1's NP_2" and does not require reordering (see Figure 6.1 (a)). However, based on the linguistic environment (or context) in which it is embedded, this Chinese possessive structure can express more sophisticated relationships which are inappropriate for the "NP_1's NP_2" expression, for example, the "NP_2 of NP_1" sequence that requires phrase swapping (see Figure 6.1 (b)). Indeed, the fluency of machine translation can be greatly improved by obtaining a more refined word order in the target language.

As discussed in Section 2.4.4, handling such phrase movements is a computationally expensive problem. It involves three elements

1. The formulation of phrase movements.
2. The linguistic features for characterising phrase reordering.
3. The agent for learning the weight parameters $\{\mathbf{w}\}_{o \in \mathcal{O}}$.

For the first task, a practical solution is to adopt the idea of predicting phrase reordering orientations, in which case the reordering distance space is reduced to a finite set of orientation classes. For the second, a variety of linguistic features such as context features (e.g. word sequences), shallow syntactic features (e.g. part-of-speech tags used in (Zens and Ney, 2006)) and statistical features (e.g. a vast amount of probabilities used in (Tillmann and Zhang, 2005)) have been utilised. Moreover, many other feature sets, such as lemma features and syntactic relationships in POS tags are also investigated, posing a feature selection problem for any learning algorithm.

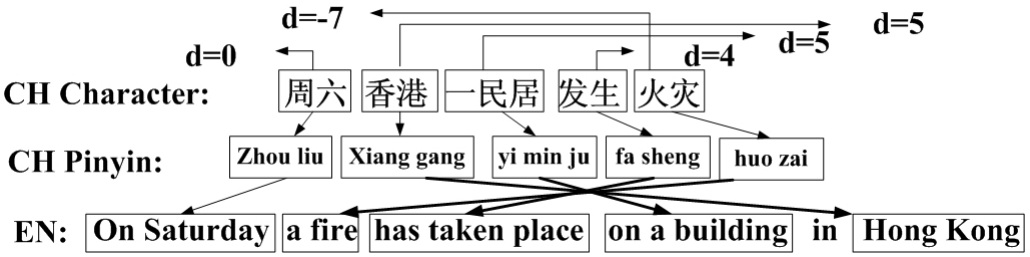


FIGURE 6.2: The phrase reordering distance d for each phrase pair in a Chinese–English sentence pair.

Instead of formulating phrase movements or investigating features sets, this chapter concentrates on the third task: exploiting a limited set of linguistic features with different learning agents. We propose a *distance phrase reordering model* (DPR) that is also inspired by the orientation prediction framework (Koehn et al., 2005). Unlike (Xiong et al., 2006; Zens and Ney, 2006) we regard phrase reordering as a classification problem and use the proposed *max-margin structure* (MMS) method to perform the classification. The method is then compared with four other learning agents – the *lexicalized reordering* (LR) model (Koehn et al., 2005), the *maximum entropy* (ME) framework (Zens and Ney, 2006), the *support vector machine* (SVM) technique and the *maximum margin regression* (MMR) approach – with the purpose of verifying its good tradeoff between the classification accuracy and the time efficiency. Furthermore, we also integrate the DPR model in a traditional SMT system; the resulting MT system is then compared with the state-of-the-art SMT system (MOSES) on several translation tasks so as to demonstrate the effectiveness of the proposed DPR model.

The remaining parts are organised as follows: a general framework of the DPR model is given in Section 6.2, which specifies the modelling of phrase movements and describes the motivations of utilising different learning agents. Then in Section 6.3 we demonstrate the linguistic features used and the training procedure for the DPR model. Section 6.4 evaluates the performance of the DPR model with both phrase reordering classification and machine translation experiments. Finally, we draw conclusions and mention areas for future work in Section 6.5.

6.2 Distance phrase reordering (DPR)

We adopt a discriminative model to capture the frequently occurring distance reorderings (e.g. Figure 6.1 (b)). An ideal model would consider every word position as a class and predict the start position of the next phrase, although in practice this is rather difficult to achieve. Hence, we consider a limited set of classes.

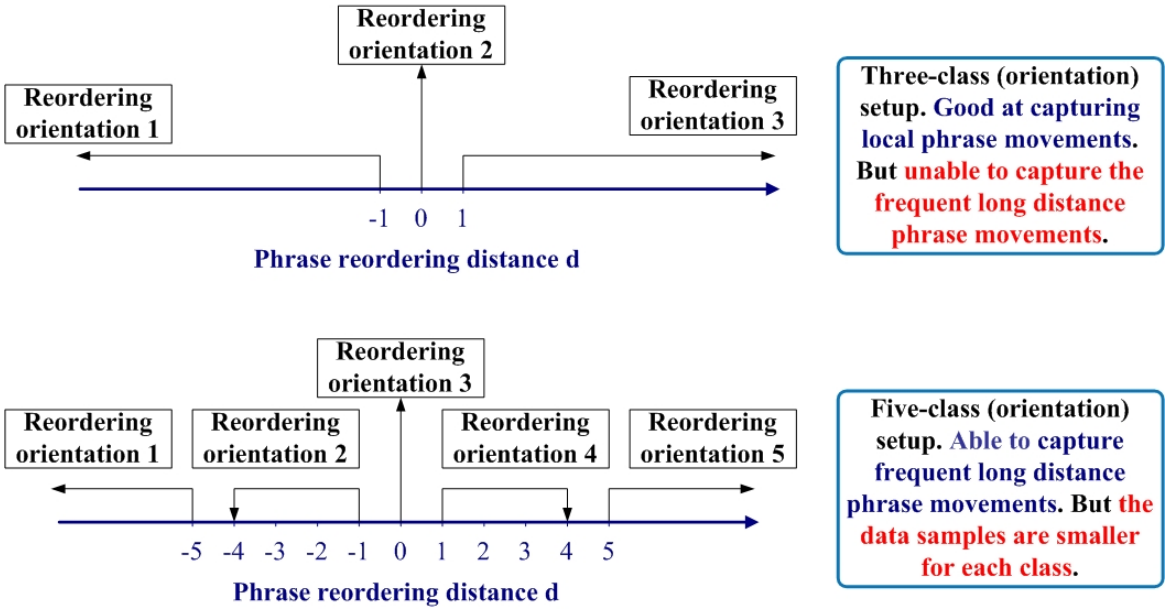


FIGURE 6.3: The phrase reordering orientations: three–class setup (top) and five–class setup (bottom).

6.2.1 Orientation class definition

Following Koehn's lexicalized reordering model, we utilise the phrase reordering distance

$$\begin{aligned} d_j \;:=\; & \text{abs}(\text{last source word position of previously translated phrase } \bar{f}_{j-1} + 1 \\ & -\text{first source word position of newly translated phrase } \bar{f}_j) \end{aligned} \tag{6.1}$$

to measure phrase movements (see Figure 6.2). The distance space $d \in \mathbb{Z}$ is then split into $C_{\mathcal{O}}$ segments (i.e. $C_{\mathcal{O}}$ classes) and the possible start positions of phrases are grouped to make up a phrase orientation set $\mathcal{O}$. Note that the more orientation classes a model has, the closer it is to the ideal model, but the smaller amount of training samples it would receive for each class. Therefore we consider two setups: a three–class approach $\mathcal{O} = \{d < 0, d = 0, d > 0\}$ and one with five classes $\mathcal{O} = \{d \leq -5, -5 < d < 0, d = 0, 0 < d < 5, d \geq 5\}$[1] (see Figure 6.3).

6.2.2 Reordering probability model and learning agents

Given a (source, target) phrase pair $(\bar{f}_j^n, \bar{e}_i^n) \in \Upsilon$ with $\bar{f}_j = [f_{j_l}, \ldots, f_{j_r}]$ and $\bar{e}_i = [e_{i_l}, \ldots, e_{i_r}]$, the *distance phrase reordering probability* has the form

$$p_d(o|(\bar{f}_j^n, \bar{e}_i^n), \{\mathbf{w}_o\}_{o\in\mathcal{O}}) := \frac{h\big(\mathbf{w}_o^T \phi(\bar{f}_j^n, \bar{e}_i^n)\big)}{\sum_{o'\in\mathcal{O}} h\big(\mathbf{w}_{o'}^T \phi(\bar{f}_j^n, \bar{e}_i^n)\big)} \tag{6.2}$$

[1]The five-word parameter setting is designed specifically for the MT experiments (see Section 6.4), which enables each class to have similar sizes of samples.

where $\mathbf{w}_o = [w_{o,0}, \ldots, w_{o,dim(\phi)}]^T$ is the weight vector measuring features' contribution to an orientation $o \in \mathcal{O}$, ϕ is the feature vector and h is a pre-defined monotonic function.

Equation (6.2) is analogous to the well-known maximum entropy framework utilised in (Zens and Ney, 2006). In contrast to learning $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$ by maximising the entropy over all phrase pairs' orientations

$$\max_{\{\mathbf{w}_o\}_{o\in\mathcal{O}}} \Big\{ - \sum_{(\bar{f}_j^n, \bar{e}_i^n)\in\Upsilon} \sum_{o\in\mathcal{O}} p(o|\bar{f}_j^n, \bar{e}_i^n, \{\mathbf{w}_o\}) \log p(o|\bar{f}_j^n, \bar{e}_i^n, \{\mathbf{w}_o\}) \Big\},$$

we propose using *maximum margin classifiers* to learn $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$. Under this framework, three discriminative models are considered, for different purposes of capturing phrase movements. We now describe each of these in the following subsections.

6.2.2.1 Support vector machine (SVM) learning

Support vector machine (SVM) is a classification technique which has shown good performance in many diverse application areas. The basic SVM is a binary classifier, hence we learn each $\mathbf{w}_o$ with a separated SVM that solves the following optimisation problem

$$\begin{array}{ll} \min\limits_{\mathbf{w}_o, \boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}_o^T\mathbf{w}_o + C \sum\limits_{(\bar{f}^n, \bar{e}^n)\in\Upsilon} \xi(\bar{f}^n, \bar{e}^n) \\ s.t. & \varphi(o_n, o)\big(\mathbf{w}_o^T \phi(\bar{f}^n, \bar{e}^n)\big) \geq 1 - \xi(\bar{f}^n, \bar{e}^n) \\ & \xi(\bar{f}^n, \bar{e}^n) \geq 0 \quad \forall (\bar{f}^n, \bar{e}^n) \in \Upsilon. \end{array} \tag{6.3}$$

where $\varphi(o_n, o)$ is an embedding function for the phrase reordering orientation o_n

$$\varphi(o_n, o) = \begin{cases} 1 & \text{if } o_n = o \\ -1 & \text{otherwise} \end{cases}$$

This approach has been successfully used for many prediction tasks. However, for N training examples (phrase pairs) the computational complexity of the SVM model is somewhere between $O(C_{\mathcal{O}}N + N^2 dim(\phi))$ and $O(C_{\mathcal{O}}N^2 + N^2 dim(\phi))$. The dependence on $C_{\mathcal{O}}$ may cause computational problems, especially when the number of phrase reordering orientations increases.

6.2.2.2 Maximum margin regression (MMR) learning

A good agent for learning $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$ should adapt to the number of phrase reordering orientations $C_{\mathcal{O}}$, enabling equation (6.2) to extend to more classes in the future. In this sense, we consider the maximum margin regression (MMR) technique, that acquires

$\{\mathbf{w}_o\}_{o\in\mathcal{O}}$ by solving the following optimisation problem (Szedmak et al., 2006)

$$\begin{aligned} \min_{\{\mathbf{w}_o\}_{o\in\mathcal{O}}} \quad & \tfrac{1}{2}\sum_{o\in\mathcal{O}}\mathbf{w}_o^T\mathbf{w}_o + C\sum_{(\bar{f}^n,\bar{e}^n)\in\Upsilon}\xi(\bar{f}^n,\bar{e}^n) \\ s.t. \quad & \sum_{o\in\mathcal{O}}\varphi(o_n,o)\mathbf{w}_o^T\phi(\bar{f}^n,\bar{e}^n) \geq 1-\xi(\bar{f}^n,\bar{e}^n), \\ & \xi(\bar{f}^n,\bar{e}^n)\geq 0, \quad \forall(\bar{f}^n,\bar{e}^n)\in\Upsilon. \end{aligned} \tag{6.4}$$

where $\varphi(o_n,o)$ is an indicator function

$$\varphi(o_n,o)=\begin{cases} 1 & \text{if } o_n=o \\ 0 & \text{otherwise} \end{cases}$$

The computational complexity of MMR is the complexity of a binary SVM (Szedmak et al., 2006), which is independent of the output structure (i.e. the number of classes). This allows the orientation class approach presented here to be extended, say to tree structured models, whilst not increasing the computational complexity. Furthermore, it allows the use of non-linear functions, going beyond the approach presented in (Zens and Ney, 2006), and is expected to provide more flexibility in the expression of phrase features.

To solve problem (6.4), we used the online version of MMR – *vector perceptron* algorithm, which is described in Table 6.2. A dual version of vector perceptron algorithm is described in Table 6.3, that allows the use of the input kernel $K^{\phi}(n,m) = \langle\phi(\bar{f}_j^n,\bar{e}_i^n),\phi(\bar{f}_{j'}^m,\bar{e}_{i'}^m)\rangle$ and the output kernel $K^{\varphi}(n,m)=\langle\varphi(o_n,o),\varphi(o_m,o)\rangle=\varphi(o_n,o_m)$.

Input of the learner: The samples $\left\{o_n,\phi(\bar{f}^n,\bar{e}^n)\right\}_{n=1}^N$, learning rate v
Initialisation: $t=0$; $\mathbf{w}_{o,t}=\mathbf{0}\quad\forall o\in\mathcal{O}$;
Repeat
  for $n=1,2,\ldots,N$ **do**
    read input: $\phi(\bar{f}^n,\bar{e}^n)$;
    if $\sum_{o\in\mathcal{O}}\varphi(o_n,o)\mathbf{w}_{o,t}^T\phi(\bar{f}^n,\bar{e}^n)<1$ **then**
      $\mathbf{w}_{o,t+1}=\mathbf{w}_{o,t}+v\varphi(o_n,o)\phi(\bar{f}^n,\bar{e}^n)^T$
      t=t+1
until converge
Output of the learner: $\mathbf{w}_{o,t+1}\in\mathbb{R}^{dim(\phi)}\quad\forall o\in\mathcal{O}$

TABLE 6.2: Pseudo-code of the *vector perceptron* algorithm (primal version).

Input of the learner: The kernels K^{ϕ} and K^{φ}, learning rate v
Initialisation: $\alpha_n = 0$, $n = 1, \ldots, N$;
Repeat
  for $n = 1, 2, \ldots, N$ **do**
    if $\sum_{m=1}^{N} \alpha_n K^{\varphi}(n,m) K^{\phi}(n,m) < 1$ **then**
      for $m = 1, 2, \ldots, N$ do
        $\alpha_m = \alpha_m + v K^{\varphi}(n,m) K^{\phi}(n,m)$
until converge
Output of the learner: (α_n), $n = 1, \ldots, N$

TABLE 6.3: Pseudo-code of the *vector perceptron* algorithm (dual version).

6.2.2.3 Max-margin structure (MMS) learning

The two techniques above only consider a fixed margin to separate one orientation class from the others. However, as the phrase reordering orientations tend to be interdependent, introducing flexible margins to separate different orientations sounds more reasonable. Take the five–class setup for example, if an example in class $d \leq -5$ is classified in class $-5 < d < 5$, intuitively the loss should be smaller than when it is classified in class $d > 5$. Therefore, learning $\{\mathbf{w}_o\}_{o \in \mathcal{O}}$ is more than a multi–class classification problem: the output (orientation) domain has an inherent structure and the model should respect it. By this motivation, we introduce the max-margin learning framework proposed in (Taskar et al., 2003) which is equivalent to minimising the sum of all classification errors

$$\min_{\{\mathbf{w}_o\}_{o \in \mathcal{O}}} \frac{1}{N} \sum_{n=1}^{N} \rho(o_n, \bar{f}^n, \bar{e}^n, \{\mathbf{w}_o\}_{o \in \mathcal{O}}) + \frac{\lambda}{2} \sum_{o \in \mathcal{O}} \|\mathbf{w}_o\|^2 \tag{6.5}$$

where $\lambda \geq 0$ is a regularisation parameter,

$$\rho(o_n, \bar{f}^n, \bar{e}^n, \{\mathbf{w}_o\}_{o \in \mathcal{O}}) = \max\left\{0, \max_{o' \neq o_n} [\triangle(o_n, o') + \mathbf{w}_{o'}^T \phi(\bar{f}^n, \bar{e}^n)] - \mathbf{w}_{o_n}^T \phi(\bar{f}^n, \bar{e}^n)\right\} \tag{6.6}$$

is a structured margin loss and function $\triangle(o_n, o')$ is applied to measure the "distance" between a pseudo-orientation o' and the correct one o_n. In the experiments, the distance matrix is pre-defined as

$$\triangle(o_n, o') = \begin{cases} 0 & \text{if } o' = o_n \\ 0.5 & \text{if } o' \text{ and } o_n \text{ are close in } \mathcal{O} \\ 1 & \text{else} \end{cases} \tag{6.7}$$

As shown in Figure 6.4, this is equivalent to constructing a heuristic tree structure in the orientation domain.

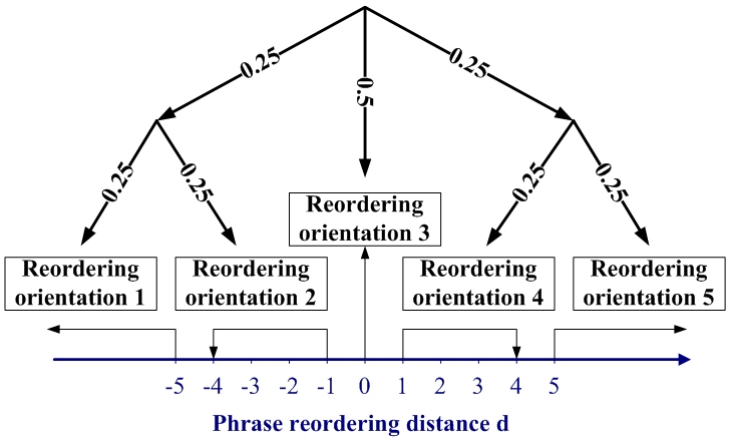


FIGURE 6.4: The heuristic tree structure constructed by the distance matrix $\triangle(o_n, o')$. There are five orientation classes (leaves) in this example.

Theoretically, the structured loss (6.6) requires that the orientation o' which is "far away" from the true orientation o_n must be classified with a large margin $\triangle(o_n, o')$, while nearby candidates are allowed to be classified with a smaller margin. This is an extension of that provided by (Collins, 2002) where no distance between classes is considered (i.e. $\triangle(o_n, o') = 1,\ \forall o'$).

Similar to the motivation discussed in Section 5.2.1, we ignored the regularisation term (i.e. $\lambda = 0$) and used a perceptron–based structured learning (PSL) algorithm to tune the parameters $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$, the pseudo-code is demonstrated in Table 6.4.

Input of the learner: The samples $\left\{o_n, \phi(\bar{f}^n, \bar{e}^n)\right\}_{n=1}^N$, learning rate η
Initialisation: $t = 0$; $\mathbf{w}_{o,t} = \mathbf{0} \quad \forall o \in \mathcal{O}$;
Repeat
  for $n = 1, 2, \ldots, N$ **do**
    $V = \max\limits_{o' \neq o_n} \left\{\triangle(o_n, o') + \mathbf{w}_{o',t}^T \phi(\bar{f}^n, \bar{e}^n)\right\}$
    $o^* = \arg\max\limits_{o' \neq o_n} \left\{\triangle(o_n, o') + \mathbf{w}_{o',t}^T \phi(\bar{f}^n, \bar{e}^n)\right\}$
    if $\mathbf{w}_{o_n,t}^T \phi(\bar{f}^n, \bar{e}^n) < V$ **then**
      $\mathbf{w}_{o,t+1}|_{o=o_n} = \mathbf{w}_{o,t}|_{o=o_n} + \eta\phi(\bar{f}^n, \bar{e}^n)$
      $\mathbf{w}_{o,t+1}|_{o=o^*} = \mathbf{w}_{o,t}|_{o=o^*} - \eta\phi(\bar{f}^n, \bar{e}^n)$
      $t = t + 1$
until converge
Output of the learner: $\mathbf{w}_{o,t+1} \in \mathbb{R}^{dim(\phi)} \quad \forall o \in \mathcal{O}$

TABLE 6.4: Pseudo-code of the *perceptron-based structured learning* (PSL) algorithm.

Table 6.4 indicates that the computational complexity of PSL is $O(Ndim(\phi)C_{\mathcal{O}})$, which still depends on the number of classes. However, compared with the previous SVM and even MMR models, PSL is substantially faster as in practice the number of classes $C_{\mathcal{O}}$ is much smaller than the number of examples N.

Notice that in PSL $\mathbf{w}_{o,t+1}$ is tested on the example $(o_n, \phi(\bar{f}^n, \bar{e}^n))$ which is not available for training $\mathbf{w}_{o,t}$, so if we can guarantee a low cumulative loss we are already guarding against over-fitting. If one wished to add regularisation to further guard against over-fitting, one could apply methods such as ALMA (Gentile, 2001) or NORMA (Kivinen et al., 2004). However, the requirement of normalising $\mathbf{w}_o$ at each step makes the implementation intractable for a large structured learning problem. As an alternative, the risk function (6.5) can be reformulated as a joint convex optimisation problem

$$\min_{\{\|\mathbf{w}_o\| \leq R\}} \max_{\{\mathbf{z}_o \in \mathcal{Z}\}} L(\{\mathbf{w}_o\}_{o \in \mathcal{O}}, \{\mathbf{z}_o\}_{o \in \mathcal{O}}) \tag{6.8}$$

with

$$\begin{aligned} L(\{\mathbf{w}_o\}_{o \in \mathcal{O}}, \{\mathbf{z}_o\}_{o \in \mathcal{O}}) = & \sum_{n=1}^{N} \max\Big\{0, \sum_{o \in \mathcal{O}} z_o^n \big(\triangle(o_n, o) + \mathbf{w}_o^T \phi(\bar{f}^n, \bar{e}^n)\big)\Big\} \\ s.t. \quad & \begin{cases} z_o^n = -1 & o = o_n \\ z_o^n \geq 0 & o \neq o_n \quad n = 1, \ldots, N \\ \sum_{o \in \mathcal{O}} z_o^n = 0 \end{cases} \end{aligned} \tag{6.9}$$

This min–max problem can then be solved by the *extra–gradient* algorithm, which is reviewed in Section 3.5.4.

6.3 Feature extraction and application

6.3.1 Feature extraction

	features for source phrase $\bar{f}_j$	features for target phrase $\bar{e}_i$
Context features	Word n–grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$	Word n–grams (subphrases) of the target phrase $[e_{i_l}, \ldots, e_{i_r}]$
Syntactic features	Word-class n-grams within a window (length d) around the source phrase edge $[j_l]$ and $[j_r]$	Word-class n-grams (subphrases) of the target phrase $[e_{i_l}, \ldots, e_{i_r}]$

TABLE 6.5: Features extracted from the phrase environment. n-gram indicates a word sequence of length n.

Following (Vickrey et al., 2005; Zens and Ney, 2006), we consider different kinds of information extracted from the phrase environment (see Table 6.5). To capture unknown

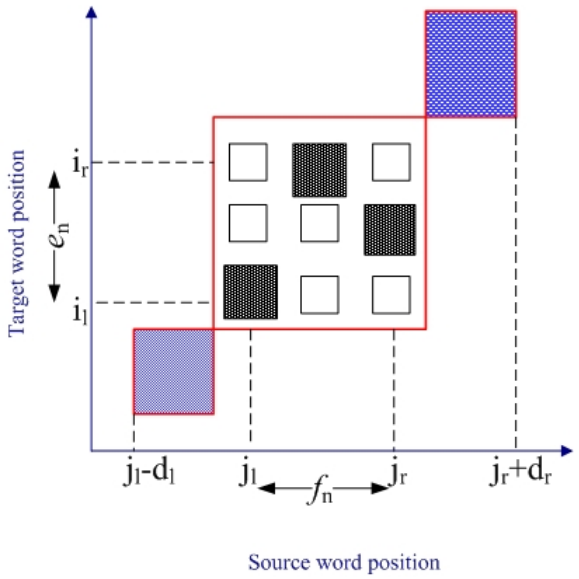


FIGURE 6.5: Illustration of the phrase pair $(\bar{f}_j^n, \bar{e}_i^n)$ (the word alignments are in black rectangle). The linguistic features are extracted from the target phrase and a window environment (blue shadow boxes) around the source phrase.

grammars and syntactic structures, some of the features would depend on the word-class[2] or part-of-speech (POS) information. Mathematically, given a sequence s from the feature environment (e.g. $s = [f_{j_l - d_l}, \ldots, f_{j_l}]$ in Figure 6.5), the features extracted are of the form

$$\phi_u(s_p^{|u|}) = \delta(s_p^{|u|}, u), \tag{6.10}$$

with the indicator function $\delta(\cdot,\cdot)$, $p = \{j_l - d_l, \ldots, j_l, j_r, \ldots, j_r + d_r\}$ and string $s_p^{|u|} = [f_p, \ldots, f_{p+|u|}]$. In this way, the phrase features are distinguished by both the content u and its start position p.

This *position-dependent linguistic* feature expression creates a very high dimensional feature space where each example $(\bar{f}_j^n, \bar{e}_i^n)$ is assigned a sparse feature vector. Figure 6.6 shows the context feature space created for all five phrase pairs in Figure 6.2 and the non-zero features for the phrase pair ("Xiang gang", "Hong Kong"). The whole feature space contains 180 features and only 9 features are non-zero for this phrase pair. The advantage of this feature expression is the collection of comprehensive linguistic information which may relate to phrase movements. However, the side effect it brings in is a large set of free parameters which may cause over-fitting on the training data.

Context(source)/Position	-3	-2	-1	1	2	3
Zhou	0	1	0	0	0	0
liu	0	0	1	0	0	0
xiang	0	0	0	0	0	0
gang	0	0	0	0	0	0
yi	0	0	0	1	0	0
min	0	0	0	0	1	0
ju	0	0	0	0	0	1
fa	0	0	0	0	0	0
sheng	0	0	0	0	0	0
huo	0	0	0	0	0	0
zai	0	0	0	0	0	0
Zhou liu	0	1	0	0	0	0
liu Xiang	0	0	0	0	0	0
Xiang gang	0	0	0	0	0	0
gang yi	0	0	0	0	0	0
yi min	0	0	0	1	0	0
min ju	0	0	0	0	1	0
ju fa	0	0	0	0	0	0
fa sheng	0	0	0	0	0	0
sheng huo	0	0	0	0	0	0
huo zai	0	0	0	0	0	0
Zhou liu Xiang	0	0	0	0	0	0
liu Xiang gang	0	0	0	0	0	0
Xiang gang yi	0	0	0	0	0	0
gang yi min	0	0	0	0	0	0
yi min ju	0	0	0	1	0	0
min ju fa	0	0	0	0	0	0
ju fa sheng	0	0	0	0	0	0
fa sheng huo	0	0	0	0	0	0
sheng huo zai	0	0	0	0	0	0

FIGURE 6.6: An example of the linguistic feature space created for all phrase pairs in Figure 6.2. Note that this example only demonstrates the context features.

6.3.2 Training and application

The training samples $\{o_n, (\bar{f}^n, \bar{e}^n)\}_{n=1}^N$ for the DPR model are derived from a general phrase pair extraction procedure described in Section 2.4.2[3].

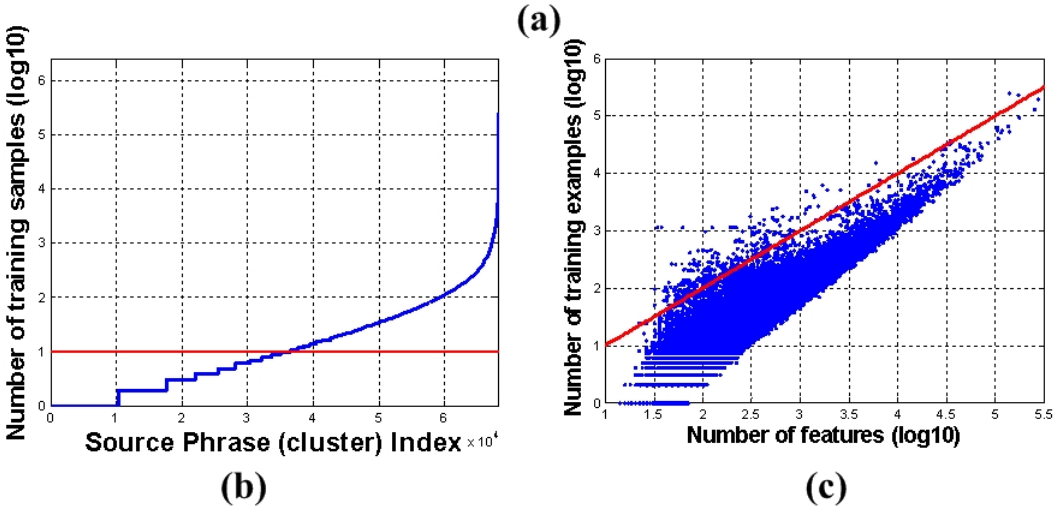
At translation time, the samples having the same source phrase $\bar{f}$ are considered to be from the same cluster (c.f. Figure 6.7 (a)). A sub-model using the above learning agents is then trained for each cluster. In our largest experiment, this framework results in training approximately 70,000 sub-DPR models (Figure 6.7 (b)). A statistics of the number of free parameters (features) against the number of training examples for each cluster is depicted in Figure 6.7 (c), implying a potential over-fitting risk. To avoid the over-fitting problem, a prior of $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$ is applied to the maximum entropy (ME) model

[2]The word-class tags are provided by the state-of-the-art SMT system (MOSES).

[3]Apart from this brief review, the readers are referred to (Koehn, 2003) for a detailed implementation of this phrase pair extraction approach.

Cluster label	安全 (Chinese character) an quan (Chinese pinyin)
Training Samples N=4	an quan – safety an quan – safely an quan – safety of an quan – safe

(a)



(b) (c)

FIGURE 6.7: (a) a cluster indicated by the source phrase "an quan" and its training samples (phrase pairs). Note that the linguistic features for the samples are not demonstrated in this example. (b) The number of training samples for each cluster (phrases are extracted from 185,000 sentence pairs). (c) The statistics of the number of features against the number of training samples (phrases are extracted from 185,000 sentence pairs).

as used in (Zens and Ney, 2006); for the MMS model, the *early stopping*[4] strategy is used which involves the careful design of the maximum number of iterations.

During the decoding, the DPR model finds the corresponding sub-model for a source phrase $\bar{f}_j$ and generates the phrase reordering probability for each orientation class with equation (6.2). In particular, for the classification experiments, the most-confident orientation is selected as the predicted class.

[4]The strategy selects the maximum number of iterations and the learning rate η by cross-validating on a validation set. In our experiments, this was done on the 185k-sentence Chinese–English MT task and the (max-iteration, learning rate) with the best performance was chosen for all other MT experiments.

6.4 Experiments

6.4.1 Corpora

Experiments used two corpora: the *parallel texts of Hong Kong laws* corpus[5] (Chinese–to–English) and the *EuroParl* corpus (French–to–English).

The parallel texts of Hong Kong laws

This bilingual *Chinese-English* corpus consists of mainly legal and documentary texts from Hong Kong which is aligned at the sentence level. The sizes of the corpus are shown in Table 6.6. As the vocabulary sizes of the corpus are very small, the content information is relatively easy to learn. However, due to many differences in word order (grammar) occurring for Chinese–English, this corpus contains many long distance phrase movements (see Figure 6.8). In this case, the phrase reordering model is expected to have more influence on the translation results, which allows us to analyse and demonstrate the effectiveness of our DPR model more straightforwardly.

Statistics	Chinese	English
Sentence Pairs	$216,250$	
Running Words	$9.76M$	$7.00M$
Vocabulary Size	$8,129$	$23,025$

TABLE 6.6: The data statistics of the Hong Kong laws corpus.

For the experiments, the sentences of lengths between 1 and 100 words were extracted and the ratio of source/target lengths was no more than 2 : 1. The training set was taken among $\{20k, 50k, 100k, 150k, 185k\}$ sentences while the test set was fixed at $1k$ sentences.

EuroParl corpus

To test the language compatibility of the DPR model, the EuroParl corpus (*French–English*) was also used. The sizes of the corpus used are shown in Table 6.7 and the statistics of phrase movements is depicted in Figure 6.9. Although the word orders between French and English are similar (represented by fewer long distance phrase movements), it is still worth predicting local phrase movements so as to improve the fluency of machine translation.

[5]The original corpus is available at `http://projects.ldc.upenn.edu/Chinese/hklaws.htm`, which however contains some sentence alignment errors. The corpus has been further cleaned up and aligned at the sentence level by the author. This refined corpus has been submitted to Linguistic Data Consortium (LDC) for publication.

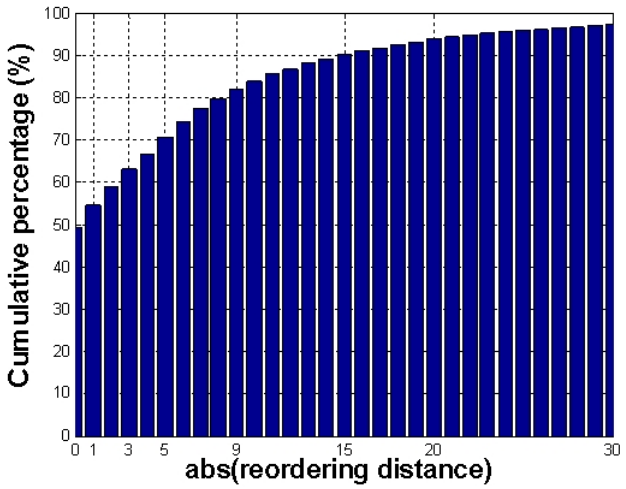


FIGURE 6.8: The statistics of phrase reordering distances d for all consistent phrase pairs (up to length 7) extracted from the *parallel texts of Hong Kong laws* corpus. The word alignments are provided by the word alignment toolkit $GIZA++$. The figure shows that short distance phrase movements (i.e. $d < 4$) only take up 62% of the whole phrase movements while very long distance phrase movements (i.e. $d \geq 15$) take up almost 10% of the whole phrase movements.

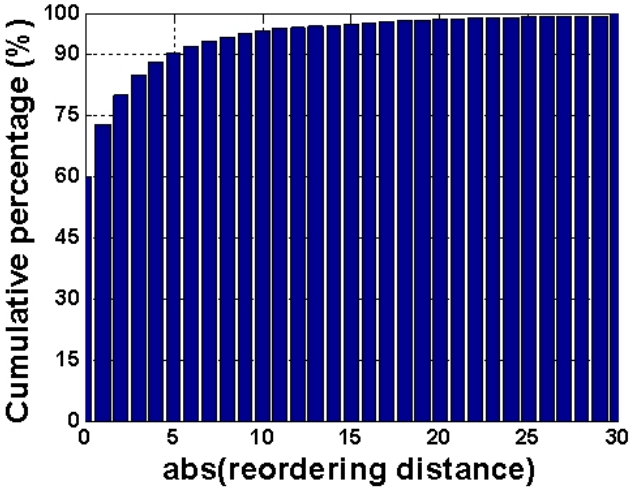


FIGURE 6.9: The statistics of phrase reordering distances d for all consistent phrase pairs (up to length 7) extracted from the *EuroParl* corpus ($50k$ sentences). The word alignments are provided by the word alignment toolkit $GIZA++$. The figure shows that short distance phrase movements (i.e. $d \leq 5$) dominate 90% of the whole phrase movements while very long distance phrase movements (i.e. $d \geq 15$) take up merely 2%.

Statistics	French	English
Sentence Pairs	50,000	
Running Words	$5.79M$	$5.20M$
Vocabulary Size	30,993	23,000

TABLE 6.7: The data statistics of the EuroParl corpus.

From the corpus, we extracted sentence pairs where both sentences had between 1 and 100 words, and where the ratio of the lengths was no more than 5 : 1. The training and the test sizes were fixed at $50k$ and $1k$ respectively.

To sum up, the Hong Kong law corpus has relatively simple content information but the phrase reordering distances are generally long; in contrast the EuroParl corpus contains complicated content information but the phrase movements are generally short.

Chinese-to-English task										
Orientations	Training set					Test set				
	$20k$	$50k$	$100k$	$150k$	$185k$	$20k$	$50k$	$100k$	$150k$	$185k$
$d < 0$	$0.17M$	$0.45M$	$0.82M$	$1.25M$	$1.63M$	$13k$	$16k$	$16k$	$17k$	$17k$
$d = 0$	$0.41M$	$1.11M$	$2.10M$	$3.30M$	$4.04M$	$28k$	$33k$	$34k$	$38k$	$38k$
$d > 0$	$0.12M$	$0.32M$	$0.61M$	$0.90M$	$1.11M$	$9k$	$10k$	$11k$	$11k$	$11k$
$d \leq -5$	$80k$	$0.20M$	$0.38M$	$0.56M$	$0.70M$	$6.0k$	$6.5k$	$7.3k$	$7.5k$	$7.4k$
$-5 < d < 0$	$90k$	$0.25M$	$0.44M$	$0.69M$	$0.83M$	$7.0k$	$9.5k$	$8.7k$	$9.5k$	$9.6k$
$d = 0$	$4.1M$	$1.11M$	$2.10M$	$3.30M$	$4.04M$	$28k$	$33k$	$34k$	$38k$	$38k$
$0 < d < 5$	$40k$	$0.10M$	$0.20M$	$0.27M$	$0.31M$	$2.5k$	$2.8k$	$2.5k$	$2.4k$	$2.2k$
$d \geq 5$	$80k$	$0.22M$	$0.41M$	$0.63M$	$0.80M$	$6.5k$	$7.2k$	$8.5k$	$8.6k$	$8.8k$

French-to-English task		
Orientations	Training set	Test set
	$50k$	$50k$
$d < 0$	$0.39M$	$14k$
$d = 0$	$0.87M$	$29k$
$d > 0$	$0.16M$	$5k$
$d \leq -5$	$77k$	$2.8k$
$-5 < d < 0$	$0.33M$	$11k$
$d = 0$	$0.91M$	$29k$
$0 < d < 5$	$0.10M$	$2.8k$
$d \geq 5$	$66k$	$2.3k$

TABLE 6.8: The training and the test sizes for three–class setup (top) and five–class setup (bottom), where "K" indicates thousand and "M" indicates million.

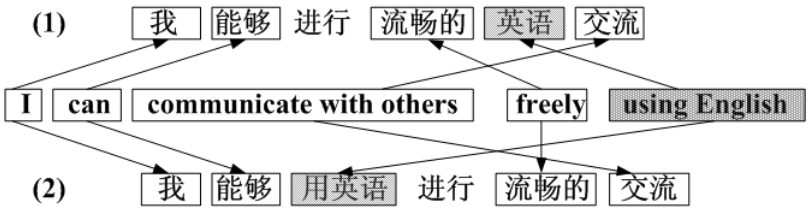


FIGURE 6.10: An English–to–Chinese example of aesthetic reordering. For the phrase "with English", although translation (1) is straightforward and clear, a translator may prefer translation (2) which makes the sentence more flowing.

6.4.2 Classification experiments

We used *GIZA++* to produce word alignments, enabling us to compare using a DPR model against a baseline LR model (Koehn et al., 2005) that uses MLE orientation prediction and a discriminative model that utilises an ME framework (Zens and Ney, 2006). In addition, we also compared the classification performance and the time efficiency among three learning agents for DPR: SVM[6], MMR and MMS, where the goal is to find the best learning agent for the MT tasks.

Two orientation classification tasks were carried out: one with three–class setup and one with five–class setup. We discarded points that had long distance reordering[7] to avoid some alignment errors caused by *GIZA++* and the aesthetic reorderings due to the translators' preference (see Figure 6.10 for example). This results in the data sizes shown in Table 6.8. The classification performance was measured by an overall precision across all orientation classes and the class-specific F1 measures and the experiments were repeated five times to access variance.

6.4.2.1 Comparison of overall precisions and the class-specific F1-scores on the Chinese–English corpus

Figure 6.11 depicts the overall precisions with respect to the training sample size, from which we observed consistent improvements for all models when the training samples increase. In addition, all discriminative models perform better than the generative LR model. The MMS approach achieves the best classification performance, with an absolute 8.5% average improvement with three-class setup and an absolute 8.7% average improvement with five classes. Similar improvements are observed when examining class-specific F1 scores on Table 6.9 and Table 6.10; the DPR model with the MMS learning agent achieves the best results. However, the DPR models with SVM and MMR techniques do not perform very well in the experiments, possibly due to the feature

[6]The multi–class SVM model is trained by SVM–Multiclass (Tsochantaridis et al., 2004).

[7]$|d| > 15$ on the Chinese-to-English data set and $|d| > 10$ on the French-to-English data set, representing less than 8% of the data.

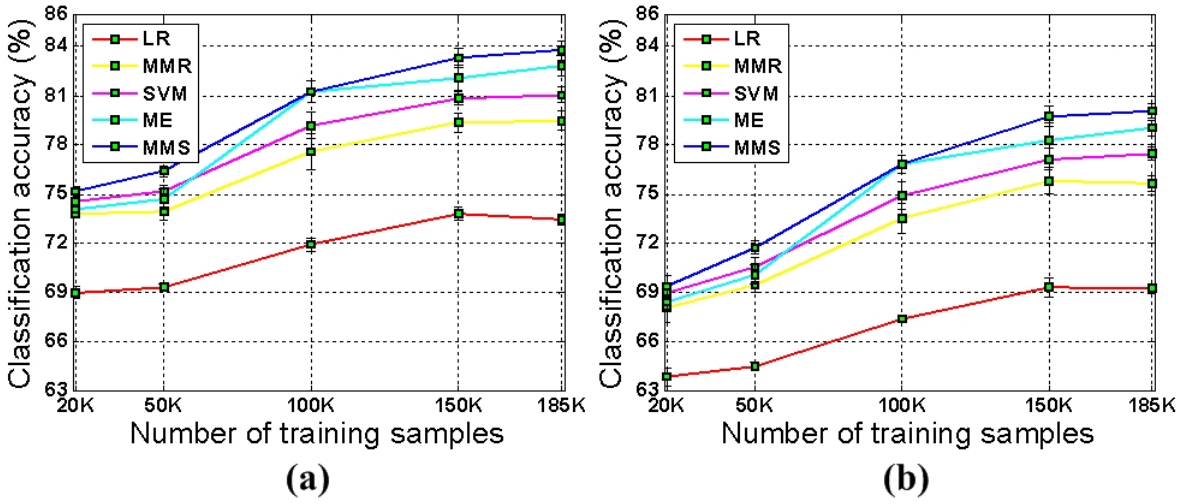


FIGURE 6.11: The overall classification precisions of three-class setup (Figure (a)) and five-class setup (Figure (b)) on the Chinese–English corpus, where "K" indicates thousand and the error bars show the variances.

expression we used. Since constructing a kernel using the sparse feature expression usually results in a very sparse kernel matrix where little similarity between samples is presented, SVM and MMR might not extract adequate information for modelling phrase movements.

When the training sample size is large, the ME model performs better than all other learning agents except MMS, showing its good ability in exploiting the features. But when the training sample size is small (e.g. $50k$–sentence task), its results are worse than that of SVM, possibly due to the over-fitting on the training data. This reveals the importance of choosing the priors for the ME models: a simple prior may not be helpful while a complicated prior usually makes the training time increase dramatically. Hence, how to choose the appropriate priors for ME in order to balance training speed and performance is often difficult. Alternatively, using the early stopping strategy DPR with MMS does not over-fit the training data, indicating that the PSL algorithm companied with early stopping already guards against over-fitting and applying a regularisation term is not necessary.

Figure 6.12 further demonstrates the average precision for each reordering distance d on the $185k$–sentence task, using the results provided by LR, ME and DPR with MMS respectively. It shows that even for long distance reorderings, the DPR model still performs well, while the LR baseline usually performs badly (more than half examples are classified incorrectly). With so many classification errors, the effect of this baseline in an SMT system is in doubt, even with a powerful language model. Meanwhile, we observed that results for forward phrase movements (i.e. $d < 0$) are better than those for backward reorderings (i.e. $d > 0$). We postulate this is because the reordering patterns for backward reorderings also depend on the orientation classes of the phrases nearby. For example, in Figure 6.2, the phrase "on a building" would be in "forward reordering"

Orientations	Training Data	Generative model	Discriminative Models			
		LR	MMR	SVM	ME	MMS
$d < 0$	$20k$	57.2 ± 0.8	63.7 ± 0.6	64.1 ± 0.9	63.9 ± 0.5	$\mathbf{64.7} \pm 0.6$
	$50k$	58.5 ± 0.1	65.6 ± 0.6	65.8 ± 0.7	65.9 ± 0.5	$\mathbf{67.4} \pm 0.1$
	$100k$	61.6 ± 1.1	69.6 ± 1.4	70.6 ± 1.3	71.8 ± 1.3	$\mathbf{74.2} \pm 0.3$
	$150k$	63.8 ± 0.6	72.3 ± 0.8	73.0 ± 0.6	75.3 ± 1.3	$\mathbf{76.5} \pm 1.0$
	$185k$	63.3 ± 0.8	72.2 ± 1.2	73.1 ± 0.8	75.7 ± 1.0	$\mathbf{76.8} \pm 1.0$
$d = 0$	$20k$	80.1 ± 0.3	83.6 ± 0.1	84.3 ± 0.2	83.7 ± 0.2	$\mathbf{84.7} \pm 0.2$
	$50k$	80.0 ± 0.1	83.4 ± 0.5	84.5 ± 0.2	84.5 ± 0.3	$\mathbf{85.5} \pm 0.2$
	$100k$	81.7 ± 0.2	85.7 ± 0.6	87.0 ± 0.3	87.8 ± 0.3	$\mathbf{88.6} \pm 0.3$
	$150k$	83.0 ± 0.3	86.8 ± 0.4	88.1 ± 0.3	89.0 ± 0.4	$\mathbf{89.9} \pm 0.4$
	$185k$	82.9 ± 0.2	86.9 ± 0.2	88.2 ± 0.3	89.5 ± 0.3	$\mathbf{90.3} \pm 0.2$
$d > 0$	$20k$	44.2 ± 0.8	55.9 ± 0.7	56.6 ± 0.8	55.6 ± 0.6	$\mathbf{58.1} \pm 1.0$
	$50k$	44.3 ± 0.3	54.9 ± 0.5	56.7 ± 0.2	56.1 ± 0.2	$\mathbf{59.3} \pm 0.5$
	$100k$	48.4 ± 2.0	63.6 ± 0.6	65.1 ± 0.2	66.5 ± 0.1	$\mathbf{68.7} \pm 0.1$
	$150k$	51.4 ± 0.6	64.7 ± 0.3	66.5 ± 0.2	68.5 ± 0.5	$\mathbf{70.8} \pm 0.3$
	$185k$	49.2 ± 1.0	64.9 ± 1.5	66.5 ± 0.3	69.6 ± 1.5	$\mathbf{71.5} \pm 1.6$

P-value in T-test				
Orientations	Generative model	Discriminative Models		
	LR	MMR	SVM	ME
$d < 0$	$1.02e-8$	$2.75e-5$	$7.27e-5$	$1.00e-4$
$d = 0$	$8.66e-10$	$8.39e-7$	$1.55e-5$	$2.19e-6$
$d > 0$	$1.97e-9$	$1.45e-6$	$7.19e-6$	$3.84e-9$

TABLE 6.9: Classification performance on the Chinese-English corpus: the class-specific F1-scores [%] for three–class setup. Bold numbers refer to the best results. P-values of T-test for statistical significance in the differences between MMS and other models are shown in the lower table.

if it does not meet another "forward" phrase "a fire has taken place". This observation shows that a richer feature set including a potential orientation class of nearby phrases may help the reordering classification and will be investigated in our future work.

6.4.2.2 Exploring DPR with MMS

With the above general view, the DPR model with the MMS learning agent has shown to be the best classifier. Here we further explore its advantages by analysing more detailed results.

Figure 6.13 first illustrates the relative improvements of DPR with MMS over LR, ME and DPRs with MMR and SVM, where we observed that the relative improvement with five-class setup is usually greater than that with three-class setup, which is especially clear for the switching orientations (i.e. $d \neq 0$). This implies the more orientation classes DPR has, the better performance MMS achieves compared with other models. The promising observation makes MMS a gold learning agent in our future work where we expect to extend the orientation set further.

Orientations	Training Data	Generative model	Discriminative Models			
		LR	MMR	SVM	ME	MMS
$d \leq -5$	$20k$	40.9 ± 2.4	46.2 ± 1.8	$\mathbf{47.2} \pm 2.4$	45.6 ± 1.9	47.0 ± 1.5
	$50k$	41.0 ± 0.2	46.5 ± 0.6	48.1 ± 0.4	47.5 ± 0.3	$\mathbf{49.6} \pm 0.7$
	$100k$	46.9 ± 0.1	54.7 ± 1.5	56.3 ± 0.8	57.3 ± 0.5	$\mathbf{58.7} \pm 0.8$
	$150k$	47.6 ± 0.9	57.1 ± 1.1	58.9 ± 1.4	60.5 ± 1.3	$\mathbf{62.1} \pm 1.1$
	$185k$	47.8 ± 0.3	57.6 ± 0.4	59.3 ± 0.3	61.8 ± 0.7	$\mathbf{63.4} \pm 0.7$
$-5 < d < 0$	$20k$	35.0 ± 1.5	44.6 ± 1.6	45.2 ± 1.3	46.6 ± 1.1	$\mathbf{47.6} \pm 1.0$
	$50k$	40.8 ± 1.5	52.3 ± 1.2	52.4 ± 0.8	53.8 ± 0.7	$\mathbf{55.5} \pm 0.2$
	$100k$	43.3 ± 0.5	55.3 ± 1.2	56.1 ± 1.5	58.7 ± 1.4	$\mathbf{60.9} \pm 0.5$
	$150k$	47.8 ± 1.7	60.8 ± 2.0	61.8 ± 2.0	65.1 ± 2.5	$\mathbf{66.1} \pm 2.0$
	$185k$	45.7 ± 1.5	59.2 ± 1.5	61.0 ± 1.5	64.8 ± 1.6	$\mathbf{66.0} \pm 1.5$
$d = 0$	$20k$	79.9 ± 0.3	83.6 ± 0.2	84.0 ± 0.2	83.9 ± 0.2	$\mathbf{84.7} \pm 0.3$
	$50k$	80.0 ± 0.1	83.7 ± 0.2	84.3 ± 0.2	84.4 ± 0.2	$\mathbf{85.5} \pm 0.2$
	$100k$	81.4 ± 0.1	86.0 ± 0.6	86.8 ± 0.5	87.6 ± 0.4	$\mathbf{88.6} \pm 0.4$
	$150k$	82.7 ± 0.3	87.2 ± 0.3	87.9 ± 0.3	88.8 ± 0.5	$\mathbf{89.8} \pm 0.3$
	$185k$	82.7 ± 0.2	87.2 ± 0.1	88.2 ± 0.2	89.5 ± 0.3	$\mathbf{90.4} \pm 0.2$
$0 < d < 5$	$20k$	13.4 ± 1.8	39.2 ± 3.0	42.8 ± 3.5	41.0 ± 3.0	$\mathbf{46.3} \pm 2.5$
	$50k$	22.0 ± 1.7	44.5 ± 1.0	47.6 ± 0.6	45.5 ± 0.4	$\mathbf{50.8} \pm 0.6$
	$100k$	19.2 ± 2.4	50.9 ± 0.9	53.6 ± 1.5	54.6 ± 1.3	$\mathbf{58.1} \pm 1.1$
	$150k$	23.8 ± 0.7	50.2 ± 0.9	54.4 ± 0.7	56.8 ± 1.7	$\mathbf{60.4} \pm 1.1$
	$185k$	19.6 ± 2.8	47.8 ± 2.8	51.4 ± 3.0	56.2 ± 3.7	$\mathbf{60.0} \pm 3.0$
$d \geq 5$	$20k$	41.4 ± 0.9	47.9 ± 3.5	50.7 ± 1.1	49.9 ± 1.0	$\mathbf{50.8} \pm 2.1$
	$50k$	39.4 ± 0.8	49.5 ± 0.2	50.9 ± 0.5	50.8 ± 0.4	$\mathbf{55.4} \pm 0.5$
	$100k$	47.0 ± 1.3	59.9 ± 0.1	61.2 ± 0.6	62.7 ± 0.6	$\mathbf{64.5} \pm 0.8$
	$150k$	48.8 ± 0.5	62.0 ± 0.1	63.8 ± 0.2	65.2 ± 0.6	$\mathbf{67.1} \pm 0.2$
	$185k$	49.4 ± 0.6	62.9 ± 1.3	64.9 ± 1.3	67.2 ± 1.4	$\mathbf{68.8} \pm 1.2$

P-value in T-test				
Orientations	Generative model	Discriminative Models		
	LR	MMR	SVM	ME
$d \leq -5$	$6.19e-7$	$3.93e-5$	$9.30e-3$	$7.89e-10$
$-5 < d < 0$	$7.77e-10$	$3.07e-7$	$3.69e-7$	$2.02e-5$
$d = 0$	$1.14e-9$	$1.19e-6$	$3.50e-6$	$8.99e-11$
$0 < d < 5$	$2.36e-11$	$5.21e-8$	$8.50e-5$	$9.51e-9$
$d \geq 5$	$4.76e-8$	$4.25e-7$	$8.56e-4$	$1.00e-3$

TABLE 6.10: Classification performance on the Chinese-English corpus: the class-specific F1-scores [%] for five–class setup. Bold numbers refer to the best results. P-values of T-test for statistical significance in the differences between MMS and other models are shown in the lower table.

We then compare the DPR model with MMS with the LR and the ME models based on the overall precision of each cluster on Figure 6.14 and Figure 6.15. Compared with the generative model LR, DPR performs better in many of the clusters, especially when given enough training samples (the black line in the figure). This verifies the advantage of the discriminative models. In particular, the larger circles which imply greater ambiguity in target translations are more often on the right; indicating MMS performs better in these ambiguous clusters, implying that the target translations also contain useful information about phrase movements.

Comparing the two discriminative models, the cluster improvement of DPR over ME

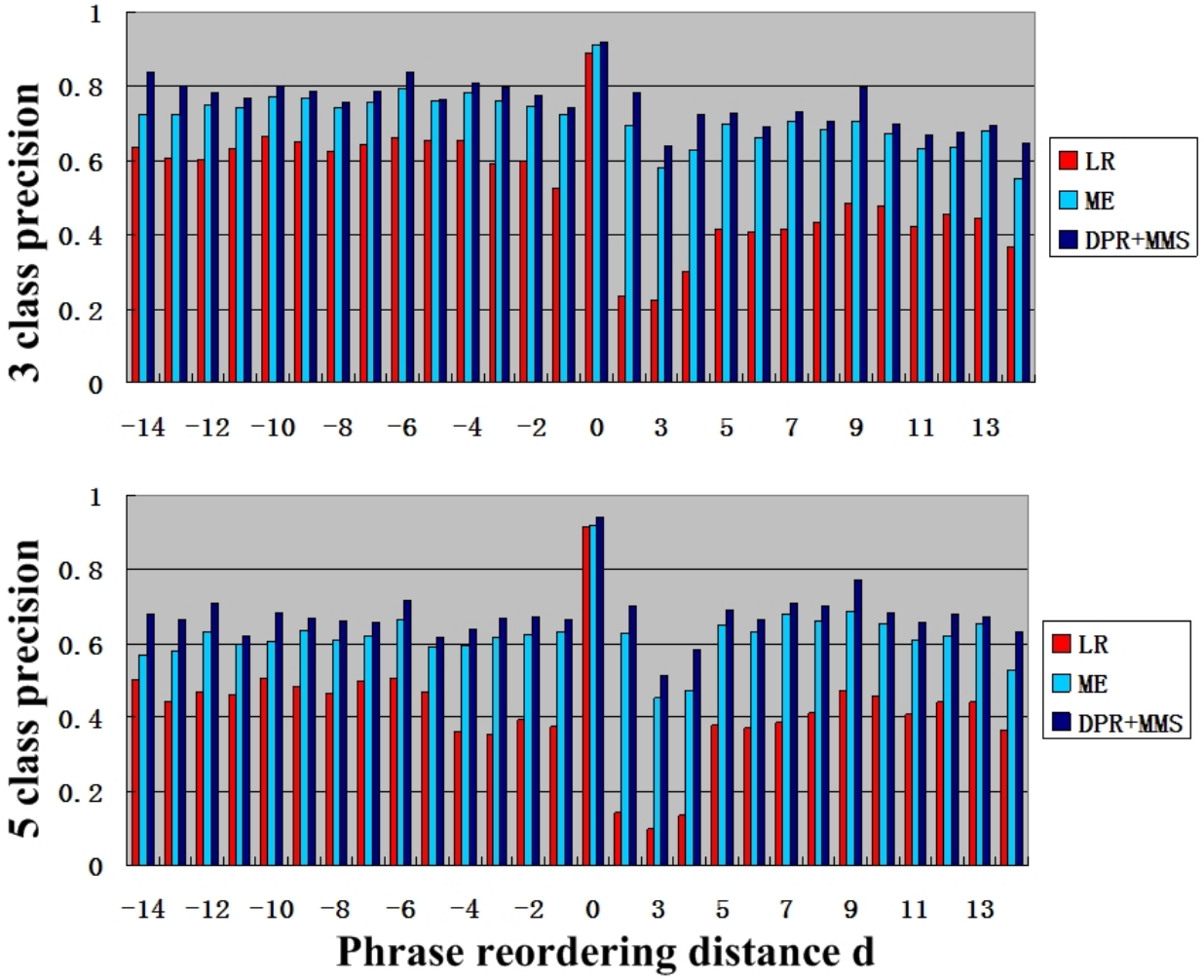


FIGURE 6.12: Classification precisions with respect to d on the $185k$–sentence task.

is smaller than that over LR, represented by the reduced number of blue circles and the increased number of red ones. However, DPR with MMS still achieves a stable improvement over ME. This is especially true when the training samples are not adequate (e.g. $50k$–sentence task), where the ME model is more likely to over-fit while the DPR with MMS still performs well.

Finally we illustrate three examples on Figure 6.16, where we observed a great improvement of DPR over LR. The first (top) example demonstrates the benefit from the target translations as by translating the Chinese source phrase "you guan" into different English words (i.e. "relating", "relates" or "relevant"), the phrase pairs usually have different but regular movements. The second (middle) example shows a grammatical structure captured by the DPR model: in English the phrase "any of" usually stays in front of the subject (or object) it modifies. In general, when given enough training samples a discriminative model such as DPR is able to capture various grammatical structures (modelled by phrase movements) better than a generative model. The final

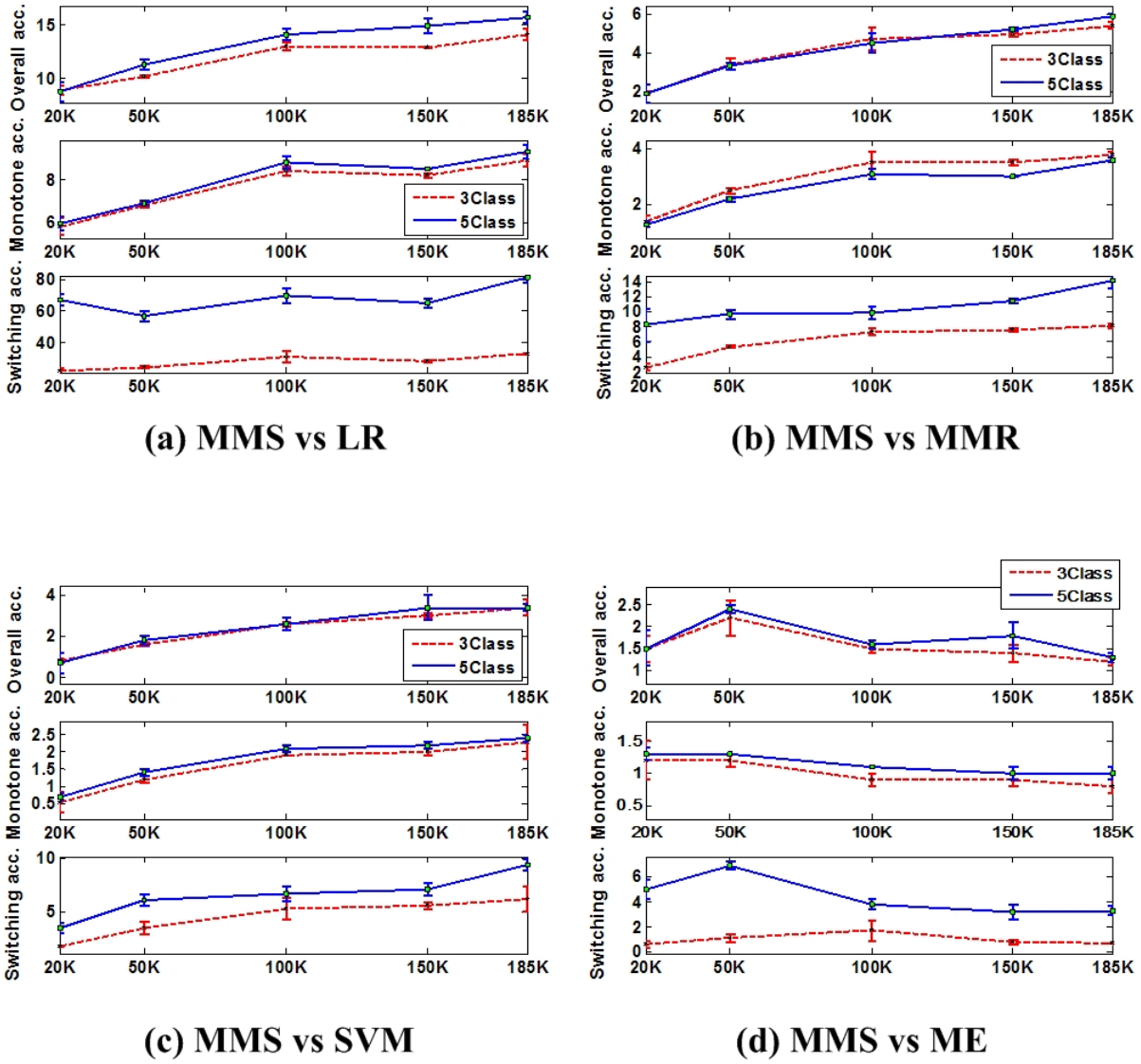


FIGURE 6.13: The average relative improvements of DPR with MMS over (a) LR, (b) DPR with MMR, (c) DPR with SVM and (d) ME for the overall precision (top), the monotone class (i.e. $d = 0$, middle) precision and the switching classes (i.e. $d \neq 0$, bottom) precision. "K" indicates thousand and the error bars show the variances.

(bottom) example depicts one type of phrase movements caused by the constant expressions in different languages (e.g. date expression). Although such expressions can be covered manually with a rule-based MT system, they can easily be captured by a DPR model as well. Hence, we conclude that the frequent phrase movements, whether caused by different grammatical structures or rule–based expressions, can be captured and the movement information is then passed on to an MT decoder to organise the target sentence structures.

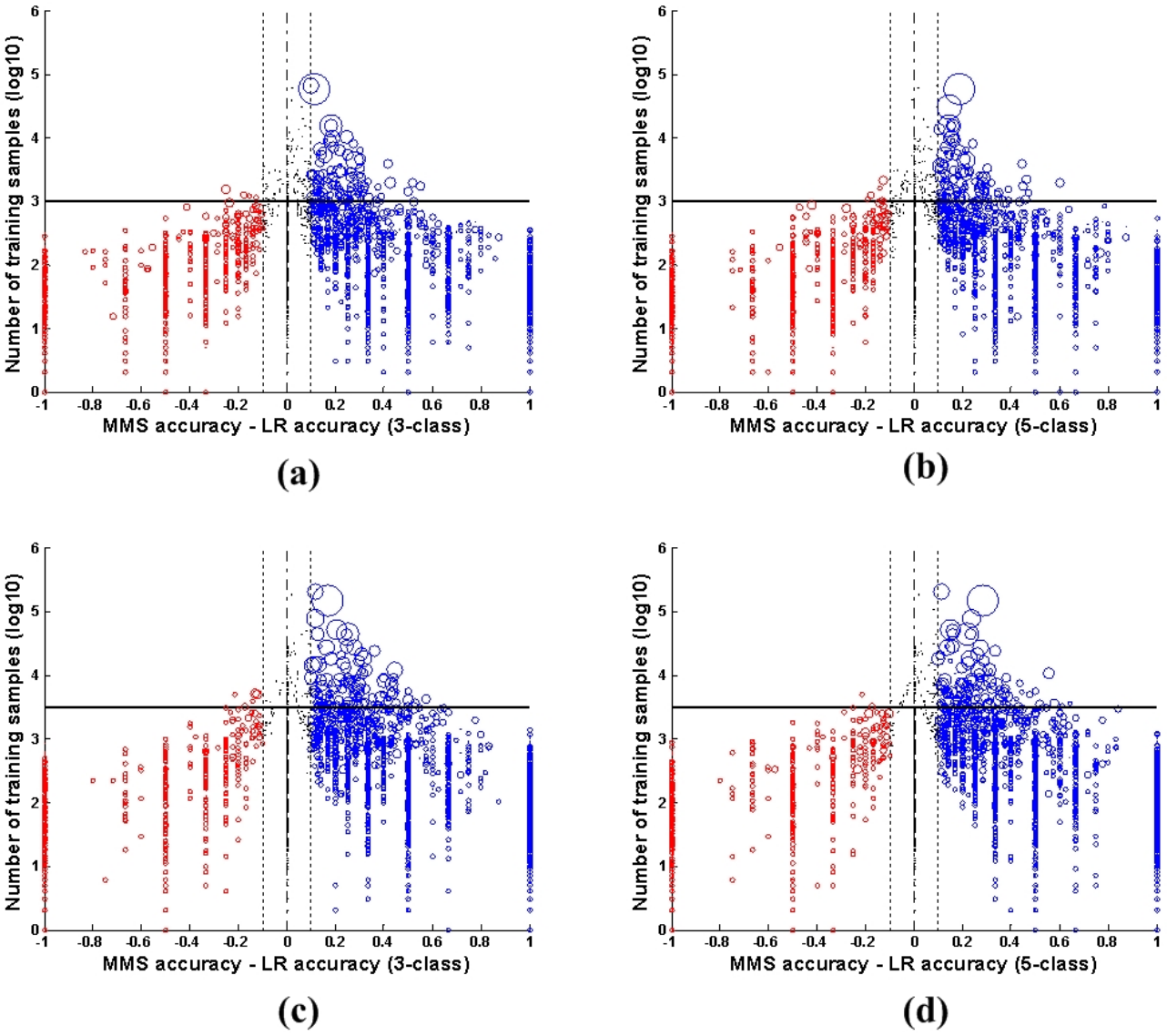


FIGURE 6.14: Scatter-plots showing the relationship between cluster precisions of DPR with MMS and LR on $50k$–sentence task (top) and $150k$–sentence task (bottom). A cluster contains all phrase pairs with a unique source phrase (e.g. Figure 6.7 (a)). Those clusters for which the performance difference (x-axes) is greater than 0.1 are shown as circles, the areas of which are proportional to the number of target translations in them. The y-axes show the number of training samples (in log 10 form) for each cluster.

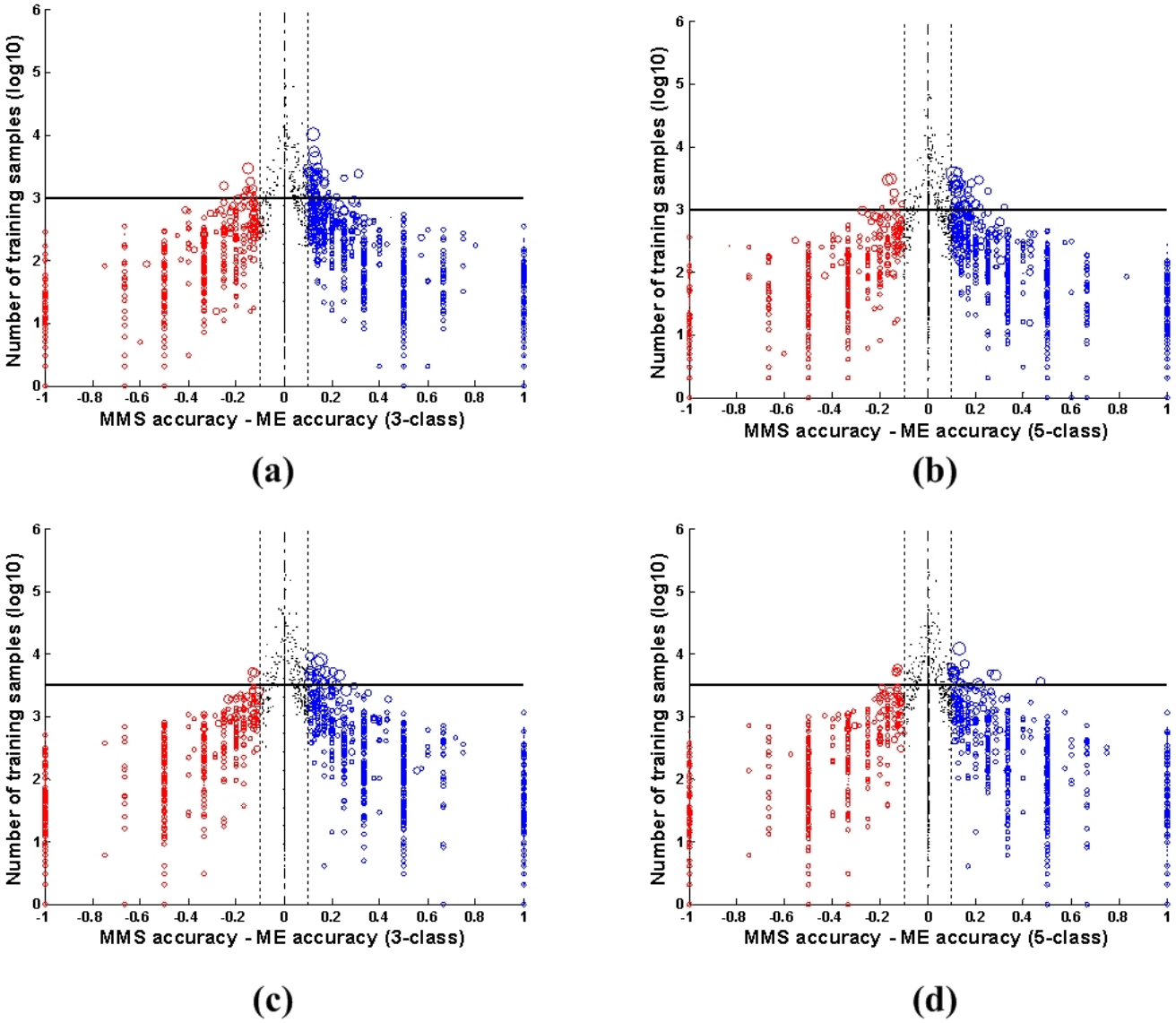


FIGURE 6.15: Scatter-plots showing the relationship between cluster precisions of DPR with MMS and ME on $50k$–sentence task (top) and $150k$–sentence task (bottom). A cluster contains all phrase pairs with a unique source phrase (e.g. Figure 6.7 (a)). Those clusters for which the performance difference (x-axes) is greater than 0.1 are shown as circles, the areas of which are proportional to the number of target translations in them.

The y-axes show the number of training samples (in log 10 form) for each cluster.

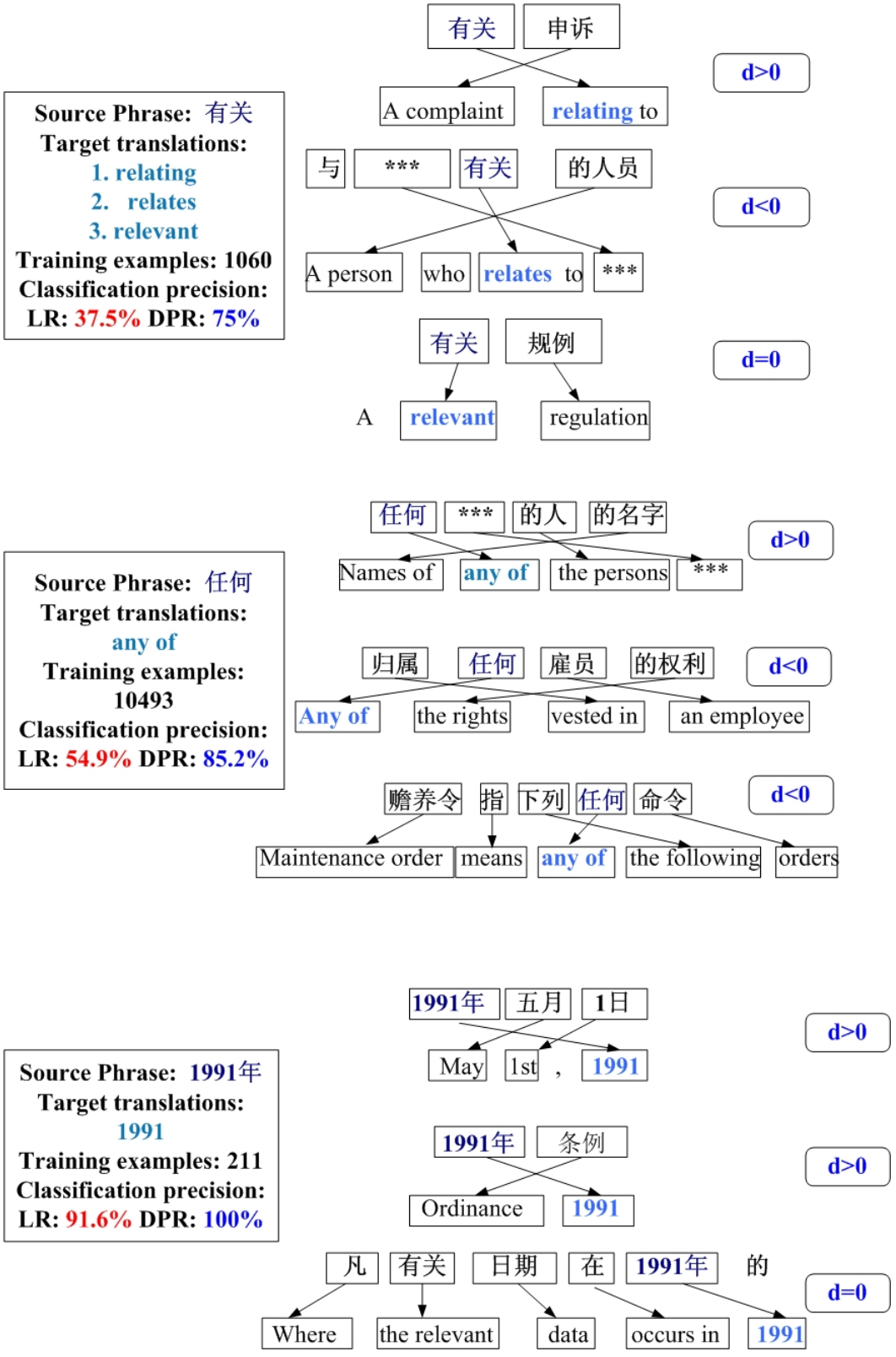


FIGURE 6.16: Phrase movements captured by the DPR model with MMS on the $50k$–sentence task.

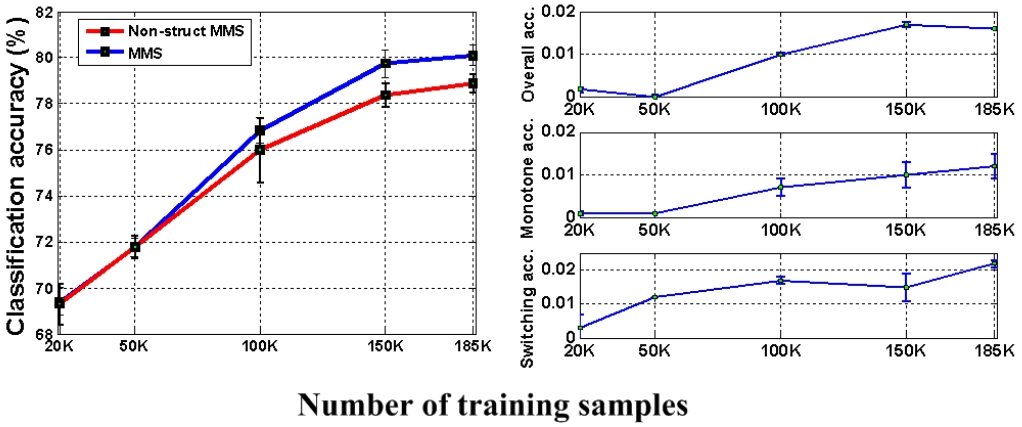


FIGURE 6.17: The overall classification precisions of the structured and the non-structured MMS algorithms (left) and the average relative improvements of the structured MMS over the non-structured MMS (right).

6.4.2.3 Structured learning versus non-structured learning

To test the effect of the pre-defined five-class orientation structure, we further compared the structured MMS with the non-structured MMS (i.e. the perceptron-based structured learning algorithm described in Section 3.4.3.6), whose results are depicted in Figure 6.17. Indeed, the orientation structure does help the classification when the training sample size is large enough. This verifies that a potential structure in the orientation domain (represented by the distance matrix) exists and the model is probable to better characterise phrase reordering if it respects this structure.

However, this heuristic structure does not always help the classification, especially when the sample size is small. We postulate this is because predictions with the pre-defined orientation structure bias towards switching orientations and thus destroy the accuracy of the monotone one, which takes a large portion (e.g. around 50%) of the test data. Therefore, how to design a more effective orientation structure, or how to learn this orientation structure automatically is a main direction of our future investigation.

6.4.2.4 A comparison of the training time

As a comparison, we plot on Figure 6.18 the training time of MMS, MMR, ME and SVM to reach the same training error tolerance[8]. For the DPR model, MMS is the fastest as expected where in contrast the SVM technique is the slowest. Moreover, training a DPR model with MMS is faster than training an ME model, especially when the number of classes increase. This is because the *Generalised Iterative Scaling* (GIS) algorithm for an ME model requires going through all samples twice at each round: one is for updating the conditional distributions $p(o|\bar{f}_j, \bar{e}_i)$ and the other is for updating $\{\mathbf{w}_o\}_{o\in\mathcal{O}}$.

[8]The MMS, MMR and ME models are coded in Python while SVM-multiclass is coded in C++.

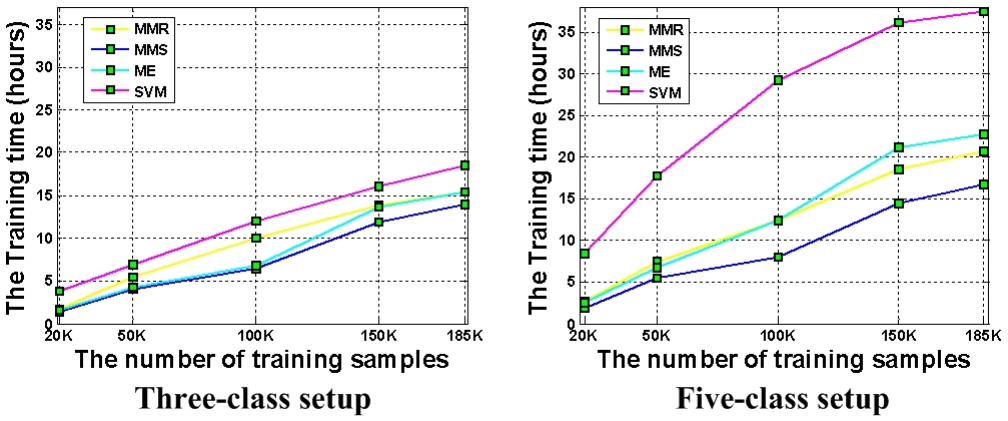


FIGURE 6.18: The training time of MMR, ME, MMS (coded in Python) and SVM (coded in C++) to reach the same training error tolerance.

Alternatively, the PSL algorithm only goes through all examples once at each round, making it faster and more applicable for larger data sets.

Scores	Generative model	Discriminative Models			
	LR	MMR	SVM	ME	MMS
$O.Acc$	61.6 ± 0.8	63.9 ± 0.5	65.6 ± 0.4	63.6 ± 0.4	$\mathbf{67.2} \pm 0.4$
$d < 0$	34.1 ± 3.7	48.2 ± 0.6	44.4 ± 0.4	47.0 ± 0.7	$\mathbf{50.0} \pm 0.5$
$d = 0$	74.2 ± 1.0	74.6 ± 0.5	76.8 ± 0.4	74.7 ± 0.3	$\mathbf{77.7} \pm 0.4$
$d > 0$	20.6 ± 1.6	32.8 ± 0.8	33.1 ± 1.0	31.7 ± 1.2	$\mathbf{37.0} \pm 1.0$

P-value in T-test				
Null hypothesis: MMS is worse than other models				
Orientations	Generative model	Discriminative Models		
	LR	MMR	SVM	ME
$O.Acc$	$4.56e-5$	$5.61e-5$	$5.70e-6$	$1.94e-6$
$d < 0$	$6.34e-4$	$4.31e-5$	$1.00e-6$	$5.95e-5$
$d = 0$	$5.05e-4$	$3.74e-5$	$1.01e-4$	$4.17e-6$
$d > 0$	$9.20e-6$	$5.19e-5$	$1.75e-5$	$1.47e-5$

TABLE 6.11: Classification performance on the French-English corpus: the overall precision ($O.Acc$) and class-specific F1-scores for three–class setup with the corresponding T-test. Bold numbers refer to the best results.

6.4.2.5 Test language compatibility: results on the French–English corpus

In order to analyse language compatibility of the DPR model, we also test it on the French–English corpus. Table 6.11 and Table 6.12 show the classification performance, suggesting that the DPR model with MMS is still the best phrase reordering model. Meanwhile, the ME model over-fits the training data and is worse than all DPR models.

However, another observation is negative. Although the word orders between French and English are similar, the phrase movements are more difficult to predict than those from the Chinese–English corpus. The reasons can be varied, which might be the increasing

Scores	Generative model	Discriminative Models			
	LR	MMR	SVM	ME	MMS
$O.Acc$	59.7 ± 0.4	62.4 ± 0.5	63.0 ± 0.5	60.9 ± 0.4	$\mathbf{64.4} \pm 0.5$
$d \leq -5$	3.7 ± 0.9	5.5 ± 0.6	7.2 ± 0.9	$\mathbf{8.9} \pm 0.9$	7.0 ± 0.7
$-5 < d < 0$	31.1 ± 0.8	42.9 ± 0.4	39.1 ± 0.5	41.3 ± 0.7	$\mathbf{46.0} \pm 0.5$
$d = 0$	74.2 ± 0.4	76.0 ± 0.4	76.9 ± 0.5	75.2 ± 0.3	$\mathbf{77.8} \pm 0.4$
$0 < d < 5$	25.6 ± 2.1	40.4 ± 1.4	42.8 ± 1.6	40.0 ± 1.4	$\mathbf{46.7} \pm 1.3$
$d \geq 5$	3.1 ± 0.9	7.1 ± 1.3	6.6 ± 1.1	7.4 ± 1.1	$\mathbf{9.0} \pm 1.0$

P-value in T-test				
Null hypothesis: MMS is worse than other models				
Orientations	Generative model	Discriminative Models		
	LR	MMR	SVM	ME
$O.Acc$	$1.90e-6$	$1.00e-4$	0.00	$3.00e-4$
$d < 0$	$5.87e-4$	$3.01e-3$	0.77	0.98
$d < 0$	$5.00e-7$	0.00	0.00	$1.00e-4$
$d = 0$	$7.00e-7$	0.00	0.00	$4.00e-4$
$d > 0$	$1.00e-6$	0.00	$2.00e-4$	$6.00e-4$
$d > 0$	$1.05e-5$	$3.00e-4$	0.00	$1.60e-3$

TABLE 6.12: Classification performance on the French-English corpus: the overall precision ($O.Acc$) and class-specific F1-scores for five–class setup with the corresponding T-test. Bold numbers refer to the best results.

alignment errors on $GIZA++$, inadequacy of the training data, a poorly designed feature set or just the increased influence of translators' aesthetic preference (e.g. the French sentence "Dans le menu scanner, choisissez propriétés." can be translated as "From the scanner menu, choose properties" or "Choose properties from the scanner menu"[9]; the former case requires phrase reordering while the latter does not).

Nevertheless, most of our observations of the DPR model are desirable properties to design an appropriate phrase reordering model, particularly the MMS technique provides a promising learning agent that has a joint-outstanding performance between the classification accuracy and the time efficiency.

6.4.3 Machine translation experiments

We now test the effect of the DPR model in an MT system, using the state-of-the-art SMT system – MOSES (Koehn et al., 2005) as the baseline system. The dotted line box in Figure 6.19 illustrates the baseline system, which consists of five sub-models:

- The phrase translation probabilities $\{p_t(\bar{f}_j|\bar{e}_i)\}$ and $\{p_t(\bar{e}_i|\bar{f}_j)\}$ derived from the relative frequency of phrase pairs.
- A lexical weight model.
- A four-gram language model.

[9]The sentence pairs come from the Xerox copy manual corpus.

- The *lexicalized reordering* (LR) model and the *word distance-based reordering* (WDR) model.
- A word penalty model and a phrase penalty model.

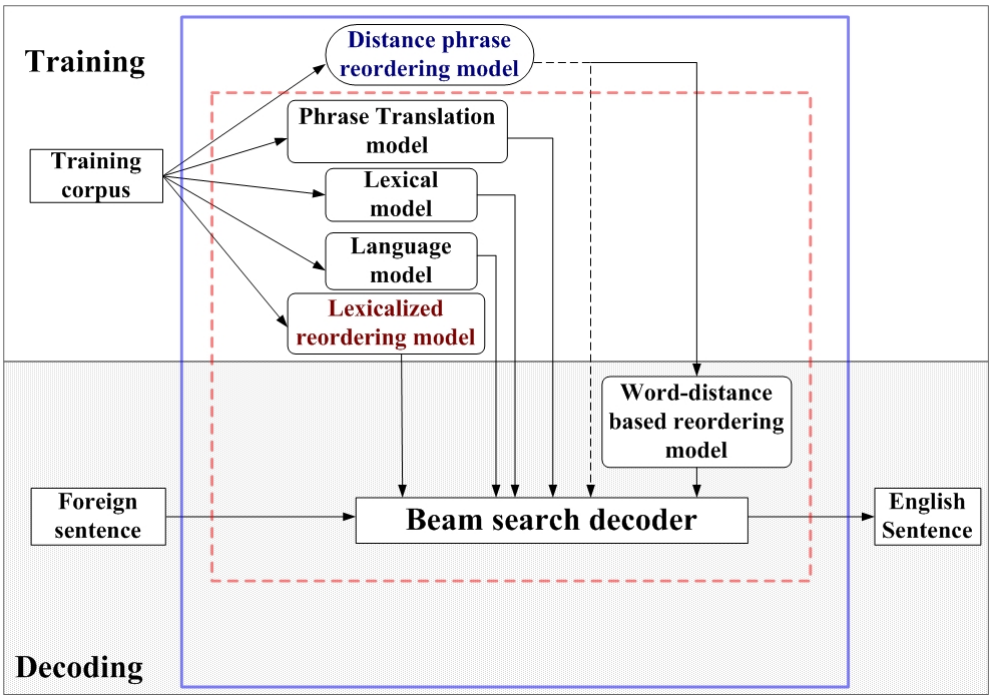


FIGURE 6.19: Training (top box) and decoding (shaded box) procedures for the baseline SMT system (dotted line box) and our MT system (solid line box). This Figure is a detailed version of Figure 2.6 which only provides a general framework of an SMT system.

To keep the comparison fair, our MT system (solid line box in Figure 6.19) just replaces MOSES's LR models with DPR while sharing all other components. In addition, we also compared the DPR model with the ME model in (Zens and Ney, 2006) on the $50k$–sentence Chinese-to-English MT task, where the results confirmed that the DPR can lead to improved performance.

According to the prominent classification performance of MMS, we chose it as the learning agent for the DPR model. In detail, all consistent phrase pairs (up to length 7) were extracted from the training sentence pairs and form the sample pool. The DPR model was then trained by the PSL algorithm and the function $h(z) = \exp(z)$ was applied to equation (6.2) to transform the prediction scores.

To make use of the phrase reordering probabilities, two strategies were applied: one is to use the probabilities directly as the reordering cost (dotted line in Figure 6.19), which is also used in (Xiong et al., 2006; Zens and Ney, 2006); the other is to use them to adjust the word distance–based reordering cost (solid line in Figure 6.19), where the reordering

cost of a sentence is computed as

$$h_o(\bar{\mathbf{e}}^I, \bar{\mathbf{f}}^I) = - \sum_{(\bar{f}_{j_m}, \bar{e}_{i_m}) \in (\bar{\mathbf{e}}^I, \bar{\mathbf{f}}^I)} \frac{d_m}{\beta p_d(o|\bar{f}_{j_m}, \bar{e}_{i_m})} \tag{6.11}$$

with tuning parameter β. Intuitively, if the DPR model has a large orientation set (i.e. the phrase movements are modelled in a precise way) and the orientation predictions are good enough, it is reasonable to use the reordering probabilities directly. However, as we experienced in Section 6.4.2, the DPR predictions with five-class setup still need improvement, especially for the switching orientations. On the other hand, if the DPR model only uses a small orientation set (e.g. three-class setup), it is able to provide very good orientation predictions. But all long distance phrase movements will have the same reordering probabilities, which may mislead the SMT decoder and spoil the translations. In this case, the distance–sensitive expression (6.11) is able to fill the deficiency of a small–class setup of DPR by penalising the long distance phrase movements. Hence in the MT experiments, we used the five-class phrase reordering probabilities directly while the three-class probabilities were used to adjust the word distance–based reordering cost.

For parameter tuning, minimum-error-rating training (MERT) (Och, 2003) was used to tune the parameters. Note that there are seven parameters which need tuning in MOSES's LR models, while there is only one for DPR. The translation performance was then evaluated by four standard MT measurements, namely word error rate (WER), BLEU, NIST and METEOR (see Section 2.6 for details).

Measure	**MOSES**	**DPR**		**ME**	
		3-class	5-class	3-class	5-class
WER [%]	$\mathbf{24.3} \pm 0.6$	24.6 ± 1.5	24.7 ± 1.1	25.3 ± 1.7	26.0 ± 2.1
BLEU [%]	44.5 ± 1.2	47.1 ± 1.3	$\mathbf{47.5} \pm 1.2$	46.17 ± 1.7	45.0 ± 2.5
NIST	8.73 ± 0.11	$\mathbf{9.04} \pm 0.26$	9.03 ± 0.32	8.72 ± 0.26	8.49 ± 0.49
METEOR [%]	66.1 ± 0.8	$\mathbf{66.4} \pm 1.1$	66.1 ± 1.1	65.0 ± 1.7	63.9 ± 2.6

TABLE 6.13: The horizontal comparison of the DPR model with the LR and the ME models on the $50k$–sentence Chinese-to-English MT task.

We first demonstrate on Table 6.13 a horizontal comparison of the DPR model with the LR and the ME models on the $50k$–sentence Chinese-to-English MT task. The improvements on most evaluations[10] over LR and ME are consistent with what are observed on the reordering classification experiments. However, the MT results show no difference between three-class setup and five-class setup, possibly due to the low classification accuracy of DPR with five-class setup (especially on the switching classes). How to improve the classification accuracy of DPR with a large class setup is hence a main challenge for our future work.

[10]The improvement on the BLUE score is significant at the 0.1 level, while the improvements on the NIST and the METEOR scores are not significant.

Since the three-class DPR achieves the same translation quality but it is faster, for the other MT tasks we only used DPR with three-class setup as the phrase reordering model. Figure 6.20 illustrates the comparison of the DPR MT system with the baseline MOSES according to the four MT evaluations, where we observed consistent improvements on most evaluations. Furthermore, the larger the sample size is, the better results DPR will achieve. This again shows the learning ability of the DPR model when given enough samples. In particular, both systems produce similar predictions in sentence content (represented by similar WERs), but our MT system does better in phrase reordering and produces more fluent translations (represented by better BLEUs).

However, if the sample size is small (e.g. the 20k-sentence task), DPR is unable to collect adequate phrase reordering information. In this case the application of DPR to an MT system is possible to mislead the MT decoder and produce low-quality translations (represented by the low qualities on WER and METEOR).

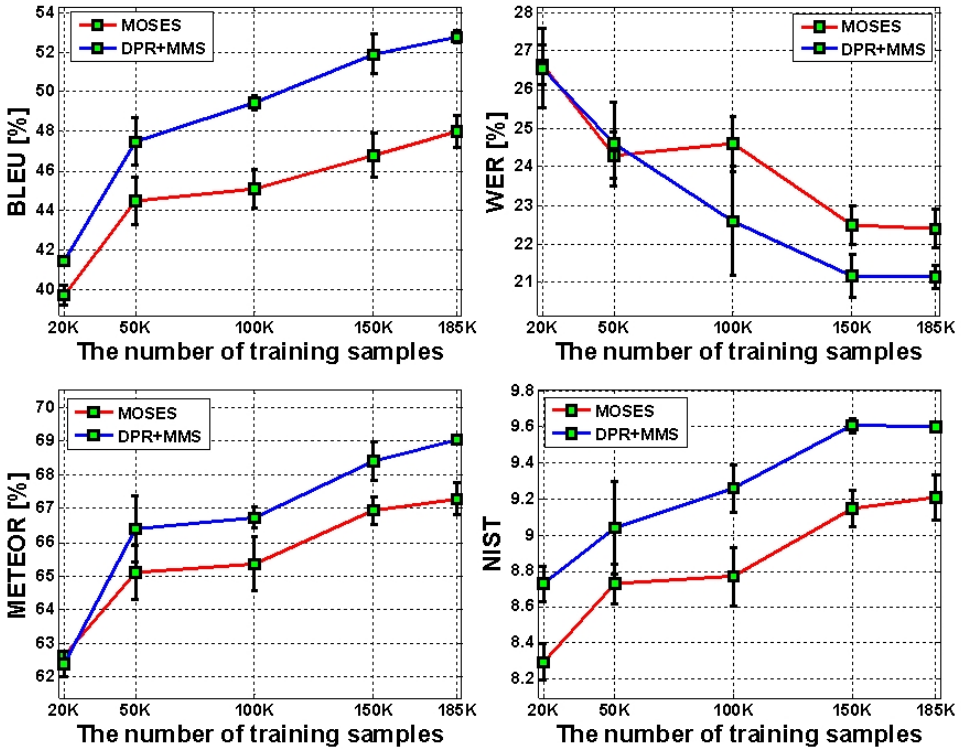


FIGURE 6.20: The translation evaluations.

Finally we demonstrate the MT evaluations on the French–English MT task on Table 6.14, using MOSES without LR, MOSES with LR and DPR respectively. Although DPR still achieves the best performance, the results are not significantly better than others. This is especially true on WER and NIST, pointing out a dilemma in tradeoff between fluency and adequacy of the translations. Meanwhile, there is little evidence showing that MOSES produces better results with LR on this corpus. This implies that a low-accurate phrase reordering model is not helpful in improving the translation qualities but a waste of computation.

Measure	**MOSES**		**DPR**
	without LR	with LR	
BLEU [%]	26.0 ± 0.2	26.3 ± 0.3	$\mathbf{28.4} \pm 4.0$
WER [%]	39.2 ± 0.4	39.2 ± 0.5	$\mathbf{38.8} \pm 2.2$
NIST	6.63 ± 0.06	6.67 ± 0.05	$\mathbf{6.77} \pm 0.42$
METEOR [%]	48.7 ± 0.3	48.8 ± 0.3	$\mathbf{50.0} \pm 1.7$

P-value in T-test		
Null hypothesis: DPR is worse than other models		
Measure	**MOSES without LR**	**MOSES with LR**
BLEU	0.13	0.17
WER	0.62	0.62
NIST	0.25	0.32
METEOR	0.11	0.13

TABLE 6.14: The comparison of the DPR model with the MOSES models on the $50k$–sentence French-to-English MT task.

6.5 Conclusion and future work

We have proposed a distance phrase reordering (DPR) model using a classification scheme and introduced a structured learning framework. The phrase reordering classification tasks have shown that DPR is better in capturing the phrase movements over the LR and the ME models, especially the MMS learning agent provides us a joint-outstanding performance between the classification accuracy and the time efficiency. In addition, compared with ME DPR with MMS is faster and hence more applicable to larger data sets. An analysis of performance confirms that the proposed MMS method is shown to perform particularly well when there is a large amount of training data, and on translation examples with large ambiguity in the target language domain.

Machine translation experiments carried out on the Chinese–English and the French–English corpora show that DPR gives more fluent translation results, which verifies its effectiveness. However, if there are not adequate training samples, the application of DPR to an MT system involves risks of destroying the translation qualities.

For future work, we aim at improving the prediction accuracy of five-class setup before applying it to an MT system, because DPR can be more powerful if it is able to provide more precise phrase position for the decoder. We also aim to formulate the phrase reordering problem as an ordinal regression problem rather than a classification proposed in this chapter. Furthermore, we will refine the learning framework of DPR by carefully designing or automatically learning the distance matrix $\triangle(o_n, o')$. Finally a richer feature set to better characterise the grammatical reorderings is also a direction of our investigation.

Chapter 7

Conclusions

7.1 Summary

The difficulties in solving the MT problems lie in formulating the complex translation modules as well as handling the high-dimensional linguistic feature space created. This thesis addresses these difficulties by means of novel machine learning techniques. It demonstrates that casting an MT problem as a machine learning problem of seeking a functional mapping produces translation accuracies that are competitive with more established statistical approaches (e.g. *maximum likelihood estimation*). In particular, a new and powerful paradigm in machine learning – structured learning, is shown to have specific promise. Taking advantages of this modern ML methodology, the thesis has demonstrated that the complex MT problems can be formulated as a *max-margin structure* (MMS) framework. In order to estimate parameters in a computationally efficient way, the work here also designs a perceptron-based learning algorithm that is shown to converge rapidly. Furthermore, the proposed framework is compared with three discriminative models: the *maximum entropy* (ME) method, the *support vector machine* (SVM) technique and the *maximum margin regression* (MMR) approach, demonstrating the effectiveness of this novel learning agent developed and applied in the thesis.

One direction of our research is to make a relatively simplistic MT system and design its sub-models with pure ML technologies. Under this motivation, we designed and constructed the *kernel–based MT system* (Chapter 4). The proposed system only consists of two components: one MMR phrase feature predictor and a Viterbi decoder. We found the performance of this implementation to be disappointingly worse than a traditional SMT system (Figure 4.9 and Figure 4.15). The individual modules, however are shown elsewhere in the literature to achieve useful performance. There are at least two reasons for this. Firstly, the MMR phrase feature predictor is not good enough to capture the content information for different phrases at the sentence level (as discussed in Section 4.5.3 and Section 4.5.4). Secondly, one single Viterbi decoder is difficult to respect all

aspects of translation, such as adequacy and fluency (as pointed out in Section 4.5.4). Hence, before solving these problems, the proposed system is not yet sufficient for the practical application. Despite the disadvantages, it is amongst the first MT systems that work with pure ML techniques; the time efficiency MMR provides, and the quality of fluency the Viterbi decoder achieves, can be helpful for the subsequent MT developers.

An alternative approach that makes use of sub-models available in other state-of-the-art MT systems and using machine learning as a tool for solving a sub problem of machine translation – *phrase feature prediction* – was implemented and evaluated (Chapter 5). We have presented a novel *max-margin structure* (MMS) approach for predicting phrase translations, which is shown to capture structural aspects of the target language domain, leading to a demonstrable performance improvement over the state-of-the-art SMT system (MOSES) on two MT tasks (Table 5.11). Furthermore, we compared its prediction accuracy and time efficiency with SVMs, where the results have shown that the proposed model is faster than SVMs without decreasing the performance in practice (as discussed in Section 5.4.1). This comparison suggests that compared with most of the other structured learning methodologies discussed whose time complexities are greater than SVMs, the proposed approach is more applicable to large scale learning problems (e.g. machine translation). In addition, a detailed study of the phrase classification performance (Figure 5.6) was carried out, suggesting when the proposed model works better and where it is better to apply. This case study can be a good guide for both MT and ML developers, who want to design or extend the existing phrase translation predictor.

The structured learning approach was also shown to be effective in modelling grammatical structure in translation (Chapter 6). We have constructed a classification scheme to approximate the grammar–learning procedure (represented by *phrase reordering*) and evaluated how the ML applications could explore grammatical structure on the Chinese-to-English and the French-to-English MT tasks. Compared with other discriminative models, the proposed MMS approach provides the best performance in capturing the grammatical phrase movements (Figure 6.11), an easier way in guarding against overfitting, as well as a good compromise between the classification accuracy and the time efficiency (Figure 6.18). In particular, the case studies (Figure 6.16) demonstrate that the frequent phrase movements, whether caused by different grammatical structures or rule–based expressions, can be captured by the proposed phrase reordering model. This observation confirms the effectiveness of the presented model and encourages its further development. Apart from its outstanding behavior in classification, when the model was integrated in a traditional MT system, the resulting system outperformed the state-of-the-art SMT system (Table 6.14 and Figure 6.20). These results are sufficiently encouraging that a small project aimed at integrating the proposed model into a public domain SMT system (MOSES[1]) is now complete (Ni et al., 2010a).

[1] `http://www.statmt.org/moses/`

7.2 Future work

There are many aspects of possible extensions of our work that could be investigated. Since machine translation is still in an exploratory stage, any development in modelling the translation process can pave the way for its success. Theoretically wise it would be of particular interest to be able to extend the MMS approach to all sub-models in an MT system, resulting in a translation system with pure ML technologies. Further research into efficient usage of linguistic features is also a challenging direction where a variety of feature selection methods can find a critical role to play.

At last we summarise the following possible aspects of future work in each of the thesis's application categories:

- For the kernel–based MT system, there are two main issues remain. First is how to scale up to large data sets. Although the *sequential minimal optimisation* (SMO) can be adapted to the solution of MMR, as mentioned in Chapter 4, the time complexity can be considerably large. In this case, the investigation on how to reformulate the optimisation problem so as to find a better tradeoff between the time and space complexities is worthwhile. The other issue is to improve the performance of the MMR phrase feature predictor. Since MMR is a kernel–based ML method, the task is equivalent to designing a more powerful and robust kernel which can better characterise the similarities between sentences.

  Further to these issues, how to integrate a language model into the system, or how to improve the Viterbi decoder to achieve an equal performance, is also of our interest.

- The proposed work on the phrase translation probability model opens the problem of how to learn the relationships between different target translations. This is equivalent to modelling the complex structures in the target language domain and requires a delicate design of the distance matrix $\Delta(c, \bar{e}_n)$. How to construct this matrix or how to learn it automatically is then a challenge at the top of the agenda. In addition, a novel perceptron-based learning agent (Shalev-Shwartz et al., 2007) that utilises a stochastic gradient descent update claimed to have faster convergence rate. This motivates our further investigation of speed improvement in the PSL algorithm by incorporating a similar update strategy to that used in (Shalev-Shwartz et al., 2007).

  When integrating the presented work into an MT system, we would like to consider a better integration framework so that the influences of the MMS–based PTP model and other models (e.g. language model) can be effectively balanced. This is possible to improve the performance in an end-to-end MT system as well as reduce the workload in the parameter-tuning procedure.

Finally the detailed study of phrase classification performance suggests using MMS when the training samples reach a reasonable size. But to predict the source phrase clusters that only contain a small amount of samples, more sophisticated ML methodologies such as multi-task learning are required.

- For the ML applications to the phrase reordering model, it is of great interest to extend the formulation of phrase reordering further, either with more orientation classes or with an ordinal regression framework. However, this extension will result in a severe problem of data sparseness, where a great effort is required to tackle this situation.

  Another interesting avenue is the modelling of the reordering distance matrix $\Delta(o_n, o')$. A further investigation of this idea may lead to a novel machine learning methodology, that builds a bridge between structured learning and multi-task learning approaches.

  A richer feature set including a potential reordering orientation of nearby phrases might better characterise the grammatical phrase movements. But importing more features would also increase the workload of the learning agent. Therefore, the exploration of a richer feature set with an appropriate feature selection method becomes a challenging problem and also requests our further investigation.

Bibliography

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

A. Argyriou, R. Hauser, C. Micchelli, and M. Pontil. A dc algorithm for kernel selection. In *Proceedings of the Twenty-Third International Conference (ICML 2006)*, 2006.

D. Arnold and L. des Tombe. Basic theory and methodology in eurotra. In *S. Nirenburg (ED.) Machine translation: theoretical and methodological issues, Cambridge University Press*, pages 114–135, Cambridge, 1987.

D. J. Arnold, L. Balkan, S. Meijer, R. L. Humphreys, and L. Sadler. *Machine translation: an introductory Guide.* NCC Blackwell Ltd., London, 1994.

S. Banerjee and A. Lavie. Meteor: an automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and Summarisation*, pages 65–72, Ann Arbor, Michigan, June 2005.

K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimisation Methods and Software*, 1:23–34, 1992.

M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14(1):1–137, 2005.

A. Berger, S. D. Pietra, and V. D. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–72, March 1996.

D. P. Bertsekas. *Nonlinear programming.* Athena Scientific, 1999.

C. M. Bishop. *Pattern recognition and machine learning.* Springer, 2006.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, pages 144–152. ACM Press, 1992.

T. Brants. Tnt – a statistical part–of–speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference (ANLPS 2000)*, pages 224–231, Seattle, WA, 2000.

P. F. Brown, J. C. Lai, and R. L. Mercer. Aligning sentences in parallel corpora. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics (ACL 1991)*, pages 169–176, California, USA, 1991.

P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19: 263–311, 1993.

L. Cai and T. Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the 13th ACM International Conference on Information and Knowledge Management*, pages 8–13, Washington, D.C., USA, 2004. ACM Press, New York, NY, USA.

C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. (meta-) evaluation of machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June 2007.

M. Carpuat and D. Wu. Context-dependent phrasal translation lexicons for statistical machine translation. In *Proceedings of MT Summit XI*, Copenhagen, Denmark, 2007.

N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, New York, NY, USA, 2006.

N. Cesa-bianchi, L. Zaniboni, and M. Collins. Incremental algorithms for hierarchical classification. In *Journal of Machine Learning Research*, pages 31–54. MIT Press, 2004.

J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. Learning bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137:43–90, 2002.

D. Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 263–270, 2005.

P. R. Clarkson and R. Rosenfeld. Statistical language modeling using the cmu–cambridge toolkit. In *Proceedings of the 8th European Conference on Speech Communication and Technology*, 1997.

J. Cocke and J. Schwartz. Programming languages and their compilers: Preliminary notes. Technical report, Courant Institute of Mathematical Sciences, New York University, 1970.

P. F. Brown J. Cocke, S. D. Pietra, V. D. Pietra, F. J. elinek, R. Mercer, and P. Roossin. A statistical approach to french/english translation. In *Proceedings of the 2nd International Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages*, Carnegie-Mellon University, 1988.

J. M. Cohen. Translation. *Encyclopedia Americana*, 27:12, 1986.

M. Collins. Discriminative training methods for hidden markov models: theory and experiments with percepton algorithms. In *C. Sammut, A. G. Hoffmann (Eds.) Proceedings of the 19th International Conference on Machine Learning (ICML 2002)*, 2002.

M. Collins and B. Roark. Incremental pasing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004)*, 2004.

C. Cortes, M. Mohri, and J. Weston. A general regression technique for learning transductions. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, Bonn, Germany, 2005.

C. Cortes and V. N. Vapnik. Support–vector networks. *Machine Learning*, 20(3):273–297, 1995.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.

K. Duan and S. S. Keerthi. Which is the best multiclass svm method? an empirical study. In *Proceedings of the 6th International Workshop on Multiple Classifier Systems*, pages 278–285, 2005.

R. O. Duda and P. E. Hart. *Pattern classification and scene analysis*. John Wiley & Sons, New York, NY, 1973.

L. Dugast, J. Senellart, and P. Koehn. Statistical post-editing on systran's rule-based translation system. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 220–223, Morristown, NJ, USA, 2007. Association for Computational Linguistics.

B. S. Everitt. *The Cambridge dictionary of statistics*. Cambridge University Press, Cambridge, 1998.

C. L. Forgy. Rete: a fast algorithm for the many pattern / many object pattern match problem. *Artificial Intellignece*, 19, 1982.

J. Gao, S. R. Gunn, and J. S. Kandola. Adapting kernels by variational approach in svm. In *Proceedings of the 15th Australian Joint Conference on Artificial Intelligence*, pages 395–406, London, UK, 2002.

C. Gentile. A new approximate maximal margin classification algorithm. *Journal of Machine Learning Research*, 2:213–242, 2001.

L. Gerber and J. Yang. Systran mt dictionary development. In *Machine Translation: Past, Present, and Future: Proceedings of Machine Translation Summit VI*, pages 211–218, San Diego, CA, 1997.

J. Giménez and L. Màrquez. Context–aware discriminative phrase selection for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 159–166, Prague, June 2007.

N. Gough and A. Way. Example-based controlled translation. In *Proceedings of the 9th EAMT Workshop:"Broadening horizons of machine translation and its applications"*, pages 73–81, Valetta, Malta, 2004a.

N. Gough and A. Way. Robust large-scale ebmt with marker-based segmentation. In *Proceedings of the 10th Conference on Theoretical and Methodological Issues in Machine Translation (TMI-04)*, Baltimore, Maryland, USA, 2004b.

J. B. Greenough and G. L. Kittredge. *Words and their ways in English speech.* The Macmillan Company, New York, 1914.

D. Groves and A. Way. Hybrid example–based smt: the best of both wolds? In *Proceedings of the ACL 2005 Workshop on Building and Using Parallel Texts: Data–driven Machine Translation and Beyond*, Ann Arbor, Michigan, USA, 2005.

D. Gusfield. *Algorithms on strings, trees, and sequences: computer science and computational biology.* Cambridge University Press, New York, USA, 1997.

A. E. Hoerl and R. W. Kennard. Ridge regression: applications to nonorthogonal problems. *Technometrics*, 12:69–82, 1970.

T. Hofmann, L. Cai, and M. Ciaramita. Learning with taxonomies: classifying documents and words, 2003.

C. Hsu and C. Lin. A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, 2002.

J. Hutchins. Fifty years of the computer and translation. *Machine Translation Review*, 6:22–24, 1997.

J. Hutchins. Example-based machine translation: a review and commentary. *Machine Translation*, 19(3–4):197–211, 2005.

W. J. Hutchins. Machine translation: a brief history. In *Concise history of the language sciences: from the sumerians to the cognitivists*, pages 431–445, Pergamon, 1995.

T. Joachims. Making large–scale svm learning pratical. In *B. Schölkopf, C. Burges, A. Smola (EDs.) Advances in kernel methods – support vector learning.* MIT Press, 1999.

T. Joachims, N. Cristianini, and J. Shawe-Taylor. Composite kernels for hypertext categorisation. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2001.

H. Kaji. Hicats/je: a japanese-to-english machine translation system based on semantics. In *Machine Translation Summit*, 1987.

H. Kaji. An efficient execution method for rule-based machine translation. In *Proceedings of the 12th conference on Computational Linguistics*, pages 824–829, Morristown, NJ, USA, 1988. Association for Computational Linguistics.

N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the 16th annual ACM symposium on Theory of computing (STOC 1984)*, pages 302–311, New York, NY, USA, 1984. ACM.

W. Karush. Minima of functions of several variables with inequalities as side conditions. Master's thesis, Department of Mathematics, University of Chicago, Chicago, IL, USA, 1939.

T. Kasami. An efficient recognition and syntax analysis algorithm for context–free languages. Technical report, Air Force Cambridge Research Laboratory, Bedford, MA, 1965.

C. Kit, H. Pan, and J. J. Webster. Example-based machine translation: a new paradigm. In *Translation and Information Technology*, pages 57–78. Chinese University of HK Press, 2003.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52(8):2165–2176, 2004.

K. Knight. Decoding complexity in word replacement translation models. *Computational Linguistics*, 25(2):607–615, 1999.

P. Koehn. *Noun phrase translation.* PhD thesis, University of Southern California, Los Angeles, 2003.

P. Koehn. Pharaoh: a beam search decoder for phrase–based statistical machine translation models. In *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA 2004)*, pages 115–124, October 2004.

P. Koehn, A. Axelrod, A. B. Mayne, C. Callison-Burch, M. Osborne, and D. Talbot. Edinburgh system description for the 2005 iwslt speech translation evaluation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT 2005)*, Pittsburgh, PA, October 2005.

R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceeding of the 14th International Joint Conference on Artical Intelligence*, number 12, pages 1137–1143, 1995.

L. Kontorovich. Uniquely decodable n–gram embeddings. *Theoretical Computer Science*, 329(1-3):271–284, 2004.

G. M. Korpelevich. The extragradient method for finding saddle points and other problems. *Ekonomika i Matematicheskie Metody*, 12:747–756, 1976.

S. Kumar and W. Byrne. Local phrase reordering models for statistical machine translation. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HTL–EMNLP 2005)*, Vancouver, Canada, 2005.

J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

VI Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:707, 1966.

C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, 1999.

M. Meilă. Data centering in feature space. Technical report, University of Washington, 2002.

M. Nagao. A framework of a mechanical translation between japanese and english by analogy principle. In *Proceedings of the international NATO symposium on artificial and human intelligence*, pages 173–180, New York, NY, USA, 1984. Inc. Elsevier North-Holland.

J. Nakamura, J. Tsujii, and M. Nagao. Grammar writing system (grade) of mu-machine translation project and its characteristics. In *Proceedings of the 10th International Conference on Computational Linguistics and 22nd annual meeting on Association for Computational Linguistics*, pages 338–343, Morristown, NJ, USA, 1984. Association for Computational Linguistics.

Y. Ni, C. Chu, C. J. Saunders, and J. Ashburner. Kernel methods for fmri pattern prediction. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2008)*, Hong Kong, 2008.

Y. Ni, M. Niranjan, C. Saunders, and S. Szedmak. Distance phrase reordering for moses - user manual and code guide. Technical report, School of Electronics and Computer Science, University of Southampton, Southampton, UK, 2010a.

Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan. Handling phrase reorderings for machine translation. In *Proceedings of the joint conference of the 47th Annual Meeting of*

the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL–IJCNLP 2009), pages 241–244, Singapore, 2009a.

Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan. Structure learning for natural language processing. In *the 11th IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2009)*, France, 2009b.

Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan. The application of structured learning in natural language processing. *Machine Translation Journal, special issue on "Pushing the frontier of Statistical Machine Translation", Springer, Netherlands*, In Press, 2010b.

Y. Ni, C. Saunders, S. Szedmak, and M. Niranjan. Exploitation of machine learning techniques in modeling phrase movements for machine translation. *in submission to Journal of Machine Learning Research, special issue on "Grammar Induction, Representation of Language and Language Learning"*, 2010c.

F. J. Och. *Statistical machine translation: from single-word to alignment templates.* PhD thesis, Computer Science Department, RWTH Aachen, Germany, October 2002.

F. J. Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, Japan, September 2003.

F. J. Och and H. Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

F. J. Och, C. Tillmann, and H. Ney. Improved alignment models for statistical machine translation. In *Proceedings of the 1999 Joint SIGDAT conference on empirical methods in natural language processing and very large corpora*, pages 20–28, University of Maryland, College Park, MD, June 1999.

E. Osuna, R. Freund, and F. Girosi. Improved training algorithm for support vector machines. In *Proceedings of the 1997 IEEE Signal Processing Society Workshop*, 1997.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002)*, 2002.

J. C. Platt. Fast training of support vector machines using sequential minimal optimisation. In *B. Schölkopf, C. Burges, A. SmolaAdvances (EDs.) Kernel methods – support vector learning*, pages 185–208. MIT Press, 1999.

J. C. Platt, N. Cristianini, and J. Shawe-taylor. Large margin dags for multiclass classification. In *Advances in Neural Information Processing Systems (NIPS 2000)*, pages 547–553. MIT Press, 2000.

J. M. Ponte and B. W. Croft. A language modeling approach to information retrieval. In *Research and development in information retrieval*, pages 275–281, 1998.

J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Learning hierarchical multi-category text classification models. In *Proceedings of the 22nd International Conference on Machine learning (ICML 2005)*, pages 744–751, New York, NY, USA, 2005. ACM.

J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor. Kernel-based learning of hierarchical multilabel classification models. *The Journal of Machine Learning Research*, 7:1601–1626, 2006.

F. Sager. Ten years of machine translation design and application: from fahqt to realism. In *Translating and the computer 10*, pages 3–10, 1988.

F. Sánchez-Martínez, M. L. Forcada, and A. Way. Hybrid rule-based – example-based mt: feeding apertium with sub-sentential translation units. In *Proceedings of the thrid Workshop on Example-Based Machine Translation*, pages 11–18, Dublin, Ireland, 2009.

B. Santorini. Part–of–speech tagging guidelines for the penn treebank project. In *Technical report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania*, 1990.

C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the 15th International Conference on Machine Learning (ICML 1998)*, pages 515–521, Madison, WI, 1998.

C. Saunders, J. Shawe-Taylor, and A. Vinokourov. String kernels, fisher kernels and finite state automata. In *Proceedings of 6th Annual Conference on Neural Information Processing Systems (NIPS 2002)*, 2002.

B. Schölkopf, C. J.C. Burges, and A. J. Smola. *Advances in kernel methods – support vector learning*. MIT Press, Cambridge, MA, 1999.

B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

B. Scott. The logos view. In *Proceedings of the 1st conference of the Association for Machine Translation in the Americas (AMTA 1994)*, Columbia, Maryland, USA, 1994.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: primal estimated sub-gradient solver for svm. In *Proceedings of the Twenty-Fourth International Conference (ICML 2007)*, Corvalis, Oregon, USA, 2007.

J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis*. Cambridge University Press, 2004.

J. Slocum. A survey of machine translation: its history, current status, and future prospects. *Computational Linguistics*, 11(1):1–17, 1985.

H. Somers. Review article: example-based machine translation. *Machine Translation*, 14(2):113–157, 1999.

S. M. Stigler. *The history of statistics: the measurement of uncertainty before 1900.* Belknap Press of Harvard University Press, Cambridge, MA, 1986.

A. Stolcke. Srilm - an extensible language modeling toolkit. In *J. H. L. Hansen and B. Pellom (Eds.) Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002)*, Denver, Colorado, USA, September 2002.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: maximum margin regression; multiclass and multiview learning at one-class complexity. Technical report, PASCAL, Southampton, UK, 2006.

B. Taskar, C. Guestrin, and D. Koller. Max–margin markov networks. In *S. Thrun, L. K. Saul, B. Schölkopf (Eds.) Proceedings of 7th Annual Conference on Neural Information Processing Systems (NIPS 2003)*, Vancouver, Canada, 2003.

B. Taskar, S. Lacoste-Julien, and M. I. Jordan. Structured prediction, dual extragradient and bregman projections. *Journal of Machine Learning Research, "Special Topic on Machine Learning and Optimization"*, pages 1627–1653, 2006.

F. E. H. Tay and L. Cao. Application of support vector machines in financial time series forecasting. *Omega*, 29(4):309–317, August 2001.

C. Tillmann. *Word re–ordering and dynamic programming based search algorithms for statistical machine translation.* PhD thesis, RWTH Aachen, Germany, 2001.

C. Tillmann. A block orientation model for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2004)*, Boston, MA, USA, 2004.

C. Tillmann, S. Vogel, H. Ney, H. Sawaf, and A. Zubiaga. Accelerated dp based search for statistical translation. In *Proceedings of the 5th European Conference on Speech Communication and Technology*, 1997.

C. Tillmann and T. Zhang. A localized prediction model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 557–564, Ann Arbor, MI, June 2005.

K. Toutanova, D. Klein, C. Manning, and Y. Singer. Feature–rich part–of–speech tagging with a cyclic dependency network. In *Proceedings of the Human Language Technology Conference and Meeting of the North American Chapter of the Association for Computational Linguistics (HLT–ACL 2003)*, pages 252–259, 2003.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Russ Greiner, Dale Schuurmans (Eds.) Proceedings of the 21st International Machine Learning Conference (ICML 2004)*. ACM Press, 2004.

J. Turian, B. Wellington, and I. D. Melamed. Scalable discriminative learning for natural language parsing and translation. In *Proceedings of the 20th Annual Conference on Neural Information Processing Systems (NIPS 2007)*, Vancouver, BC, 2007.

V. N. Vapnik. *The nature of statistical learning theory.* Inc. Springer-Verlag New York, 1995.

D. Vickrey, L. Biewald, M. Teyssier, and D. Koller. Word–sense disambiguation for machine translation. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT–EMNLP 2005)*, pages 771–778, 2005.

J. P. Vinay and J. Darbelnet. *Comparative stylistics of French and English: a methodology for translation.* John Benjamins Publishing Company, Amsterdam/Philadelphia, 1995.

S. V. N. Vishwanathan and A. J. Smola. *Fast kernels for string and tree matching*, chapter K. Tsuda and B. Schólkopf and J. Vert, (EDs.) Kernels and Bioinformatics. MA: MIT Press, 2004.

S. V. N. Vishwanathan and C. H. Teo. Fast and space efficient string kernels using suffix arrays. In *Proceedings of the 23rd International Conference on Machine learning (ICML 2006)*, Pittsburgh, PA, 2006.

Z. Wang and J. Shawe-Taylor. Large-margin structured prediction via linear programming. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS 2009)*, Clearwater Beach, Florida, USA, 2009.

Z. Wang, J. Shawe-Taylor, and S. Szedmak. Kernel regression based machine translation. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL 2007)*, pages 185–188, Rochester, New York, USA, 2007.

W. Weaver. Translation. *W. N. Locke, A. D. Booth (EDs.) Machine translation of languages: fourteen essays*, pages 15–23, 1949.

G. Willée, B. Schoröder, and H. Schmitz. Machine translation today and tomorrow. In Sankt Augustin: Gardez! Verlag, editor, *Computerlinguistik: was geht, was kommt?*, pages 159–162, 2002.

D. Wu. A polynomial-time algorithm for statistical machine translation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996)*, 1996.

D. Wu. Alignment. In *N. Indurkhya, F. J. Damerau (Eds.) Handbook of natural language processing (second edition)*, Boca Raton, Florida, USA, 2010. CRC Press, Taylor and Francis Group.

D. Xiong, Q. Liu, and S. Lin. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 521–528, Sydney, July 2006.

Daniel H. Younger. Recoginition and parsing of context–free languages in time n^3. *Information and Control*, 10(2):189–208, 1967.

R. Zens, O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. The rwth phrase–based statistical machine translation system. In *proc. of the International Workshop on Spoken Language Translation (IWSLT 2005)*, pages 155–162, Pittsburgh, PA, October 2005.

R. Zens and H. Ney. Discriminative reordering models for statistical machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 55–63, New York City, June 2006.

R. Zens, H. Ney, T. Watanabe, and E. Sumita. Reordering constraints for phrase–based statistical machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-2004)*, pages 205–211, Geneva, Switzerland, 2004.