An investigation into the impact of workflow design and aggregation on achieving quality result in crowdsourcing classification tasks

by

Qiong Bu

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

in the
Faculty of Engineering and Physical Sciences
School of Electronics and Computer Science

February 2020
Microtask crowdsourcing has been applied in many fields in the past decades, but there are still important challenges not fully addressed, especially in task/workflow design and aggregation methods to help produce a correct result or assess the quality of the result. This research took a deeper look at crowdsourcing classification tasks and explored how task and workflow design can impact the quality of the classification result. This research used a large online knowledge base and three citizen science projects as examples to investigate workflow design variations and their impacts on the quality of the classification result based on statistical, probabilistic, or machine learning models for true label inference, such that design principles can be recommended and applied in other citizen science projects or other human-computer hybrid systems to improve overall quality. It is noticeable that most of the existing research on aggregation methods to infer true labels focus on simple single-step classification though a large portion of classification tasks are not simple single-step classification. There is only limited research looking into such multiple-step classification tasks in recent years and each has a domain-specific or problem-specific focus making it difficult to be applied to other multiple-steps classifications cases. This research focused on multiple-step classification, modeling the classification task as a path searching problem in a graph, and explored alternative aggregation strategies to infer correct label paths by leveraging established individual algorithms from simple majority voting to more sophisticated algorithms like message passing, and expectation-maximisation. This research also looked at alternative workflow design to classify objects using the DBpedia entity classification as a case study and demonstrated the pros and cons of automatic, hybrid, and completely human-based workflows. As a result, it is able to provide suggestions to the task requesters for crowdsourcing classification task design and help them choose the aggregation method that will achieve a good quality result.
# Contents

**Acknowledgements**  xiii  

1 **Introduction**  1  
1.1 Motivation  3  
1.2 Research Questions  6  
1.3 Contributions  7  
1.4 Structure  8  

2 **Background**  11  
2.1 Microtask Crowdsourcing  11  
2.1.1 Basics and Characteristics of Microtask Crowdsourcing  12  
2.1.2 Task and Workflow Design  16  
2.1.3 Quality in Microtask Crowdsourcing  19  
2.1.4 Other Aspects of Microtasks Crowdsourcing  24  
2.2 Classification  28  
2.2.1 Classification Basics  28  
2.2.2 Entity Classification  30  
2.2.3 Image Classification  31  
2.3 Classification Task via Microtask Crowdsourcing  33  
2.3.1 Task Design  34  
2.3.2 Workflow Design  35  
2.4 Classification Quality Assessment  38  
2.4.1 Quality Dimensions  38  
2.4.2 Quality Metrics  40  
2.4.3 Inference Algorithms  42  

3 **Methodology**  47  
3.1 Foundations of Multiple-Step Classification  47  
3.2 Definitions  50  
3.3 Overview  51  
3.4 Scenarios for Evaluation of Research Questions  55  
3.5 Aggregation in Multiple-step Classification  56  
3.6 Classification Workflow  56  
3.7 Combining Workflow and Aggregation  58  

4 **Aggregation in Multiple-step Classification**  61  
4.1 Overview  61  
4.2 Aggregation in Multiple-step Classification  62
CONTENTS

4.2.1 Problem Formalisation .................................. 63
4.2.2 Existing Algorithms .................................... 67
  4.2.2.1 Majority Voting (MV) ................................ 68
  4.2.2.2 Expectation Maximisation (EM) ...................... 68
  4.2.2.3 Message Passing (MP) ................................. 68
4.2.3 Multiple-step Classification Aggregator .................. 70
4.2.4 Quality Assessment Metric .............................. 75
4.3 Experiments and Results ................................... 77
  4.3.1 Experiment 1: Initial Investigation of Dependency-aware Aggregation strategy vs. Path as Atomic Label Aggregation ................. 78
    4.3.1.1 Data .................................................. 78
  4.3.2 Experiment 2: Extended Investigation on Different Workflows Under Volunteer and Paid Contexts ......................... 78
    4.3.2.1 Data .................................................. 78
    4.3.2.2 Experiment Setting ................................. 79
  4.3.3 Experiment 3: Further Investigation of Multiple-step Classification Aggregations ........................................... 80
  4.3.4 Results ................................................... 81
    4.3.4.1 Experiment 1 Results ................................ 81
    4.3.4.2 Experiment 2 Results ................................ 84
    4.3.4.3 Experiment 3 Results ................................ 85
4.4 Discussion .................................................... 92

5 Classification Workflow ...................................... 97
  5.1 Overview .................................................... 97
  5.2 Classification Workflow Model in Entity Classification .......... 98
    5.2.1 Formalization .......................................... 99
    5.2.2 Predictors .............................................. 101
    5.2.3 Workflows .............................................. 102
  5.3 Applying the Model to DBpedia Entity Classification .......... 104
    5.3.1 Microtask Design ...................................... 104
    5.3.2 Implementation ...................................... 105
  5.4 Evaluation .................................................. 107
    5.4.1 Prediction Quality .................................... 108
    5.4.2 Cost .................................................. 110
    5.4.3 The Quality of Human Output .......................... 111
    5.4.4 Crowdsourcing Tasks .................................. 112
  5.5 Discussion .................................................. 112

6 Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions .................................. 117
  6.1 Overview .................................................... 117
  6.2 Hypothesis .................................................. 119
  6.3 Experiment Design ......................................... 120
    6.3.1 Design Considerations ................................ 120
    6.3.2 Task .................................................. 124
    6.3.3 Task Setting ......................................... 127
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.3.4</td>
<td>Data</td>
<td>127</td>
</tr>
<tr>
<td>6.4</td>
<td>Result</td>
<td>129</td>
</tr>
<tr>
<td>6.5</td>
<td>Discussion</td>
<td>132</td>
</tr>
<tr>
<td>7</td>
<td>Conclusions and Future Work</td>
<td>135</td>
</tr>
<tr>
<td>7.1</td>
<td>Conclusion</td>
<td>135</td>
</tr>
<tr>
<td>7.2</td>
<td>Discussion</td>
<td>140</td>
</tr>
<tr>
<td>7.3</td>
<td>Future Direction</td>
<td>142</td>
</tr>
</tbody>
</table>

**Bibliography** | 145
List of Figures

1.1 “A proposed framework for future crowd work processes to support complex and interdependent work.” (Kittur et al. (2013)) ........................................... 3

1.2 Example of a Multiple-Step Classification task from Snapshot Serengeti. The crowd is asked to choose the animal type and estimate how many animals are in the picture. Wider arrows indicate the corresponding choices are popular with the crowd. ........................................... 5

1.3 The two models developed in this research (within the blue call-outs), and where do they fit in the “Proposed framework for future crowd work processes to support complex and interdependent work.” (Kittur et al. (2013)) ........................................... 8


2.2 Mapping the IQ dimensions into the PSP/IQ model. Lee et al. (2002) .......... 40

2.3 Comparisons of different methods that address the inference problem in Crowdsourcing from Zheng et al. (2017) ......................................................... 44

3.1 Representation of a simple task with binary-options ............................... 48

3.2 Representation of a task with multiple-options ........................................ 48

3.3 Classification tasks from Zooniverse, with the number of min/max steps ...... 49

3.4 Representation of Dark Skies workflow from Cities at Night (Multiple-step Classification) ................................................................. 50

3.5 Representation of Snapshot Sergenti workflow from Zooniverse (Multiple-step Classification) ................................................................. 50

3.6 Representation of GalaxyZoo workflow from Zooniverse (Multiple-step Classification) ................................................................. 51

3.7 Normal Flow for Classification Tasks ..................................................... 52

3.8 Visionary (ideal) framework for classification tasks ............................... 53

3.9 Overall process for answer inference in Multiple-step Classification tasks ... 56

3.10 Model of classification workflow with predictor, error detector and corrector 57

3.11 Integrate both classification model and aggregation model to validate the important design factors ......................................................... 58

4.1 Applying different algorithms and aggregation strategies ....................... 64

4.2 Graph representation of an example classification workflow $W_f$ vs. corresponding classic way of looking at the classification with multiple questions ......................................................... 65

4.3 UI for GalaxyZoo job in Crowdflower ..................................................... 80

4.4 UI for Darkskies job in Crowdflower ..................................................... 81

4.5 UI for the Serengeti task in Crowdflower ................................................ 82
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6</td>
<td>Averaged DCG (Discounted Cumulative Gain) for each algorithm</td>
<td>83</td>
</tr>
<tr>
<td>4.7</td>
<td>Precision at K</td>
<td>83</td>
</tr>
<tr>
<td>4.8</td>
<td>MV votes distribution</td>
<td>84</td>
</tr>
<tr>
<td>4.9</td>
<td>Averaged DCG (Discounted Cumulative Gain) for each algorithm</td>
<td>85</td>
</tr>
<tr>
<td>4.10</td>
<td>Accuracy by node level (Serengeti)</td>
<td>87</td>
</tr>
<tr>
<td>4.11</td>
<td>Accuracy by node level (Darkskies)</td>
<td>88</td>
</tr>
<tr>
<td>4.12</td>
<td>Accuracy by node level (Galaxyzoo)</td>
<td>89</td>
</tr>
<tr>
<td>4.13</td>
<td>Average DCG for the top 20 inferred paths by all three algorithms in dependency-aware adapted implementation</td>
<td>90</td>
</tr>
<tr>
<td>5.1</td>
<td>Representation of three workflows with human participation steps highlighted in blue</td>
<td>98</td>
</tr>
<tr>
<td>5.2</td>
<td>The user interface for T1 tasks based on free text input</td>
<td>106</td>
</tr>
<tr>
<td>5.3</td>
<td>The user interface for T2 tasks based on a list of candidate options</td>
<td>106</td>
</tr>
<tr>
<td>5.4</td>
<td>Freetext: judgement number vs. matched answers</td>
<td>107</td>
</tr>
<tr>
<td>5.5</td>
<td>Overview of the entity corpus</td>
<td>108</td>
</tr>
<tr>
<td>5.6</td>
<td>Precision-recall curves for the output of the different predictors with respect to the gold standard</td>
<td>109</td>
</tr>
<tr>
<td>5.7</td>
<td>Precision at confidence level curves for the output of the different predictors with respect to the gold standard</td>
<td>110</td>
</tr>
<tr>
<td>5.8</td>
<td>The overall output quality of the workflows</td>
<td>111</td>
</tr>
<tr>
<td>5.9</td>
<td>Precision at different options list size with the $W_{hybrid}$ predictor</td>
<td>112</td>
</tr>
<tr>
<td>6.1</td>
<td>Number of species filtered by a specific filter. Note: FilterGroup and Sub-Filter are the same as the ones shown in Table 6.1.</td>
<td>121</td>
</tr>
<tr>
<td>6.2</td>
<td>Species and the corresponding classification precision</td>
<td>122</td>
</tr>
<tr>
<td>6.3</td>
<td>Correctly classified species and the corresponding number of filters used</td>
<td>123</td>
</tr>
<tr>
<td>6.4</td>
<td>Classification step 1 where existing optional filters are converted to options of a mandatory question</td>
<td>125</td>
</tr>
<tr>
<td>6.5</td>
<td>Example when horns filter/feature is chosen, corresponding options are shown</td>
<td>125</td>
</tr>
<tr>
<td>6.6</td>
<td>Showing candidates when the chosen feature can narrow down the species to less than 7 options</td>
<td>126</td>
</tr>
<tr>
<td>6.7</td>
<td>Asking user to pick another feature when currently chosen feature cannot narrow down the species to less than 7 options</td>
<td>127</td>
</tr>
<tr>
<td>6.8</td>
<td>Once a species is chosen, the description corresponding to that species is shown so that user can make sure this is the species they expect</td>
<td>128</td>
</tr>
<tr>
<td>6.9</td>
<td>User is provided with Back/Restart in all steps (the example here is when the feature user has chosen does not match any species, they can go back to pick a different feature or restart the process)</td>
<td>129</td>
</tr>
<tr>
<td>6.10</td>
<td>Accuracy by level (Species)</td>
<td>130</td>
</tr>
<tr>
<td>6.11</td>
<td>Accuracy by path</td>
<td>131</td>
</tr>
</tbody>
</table>
# List of Tables

2.1 Representational task/workflow design ........................................ 38  
2.2 Popular Inference algorithms for aggregating labels from multiple workers 46  
3.1 Definitions .................................................................................. 52  
4.1 Summary of key notations .............................................................. 66  
4.2 Notations specific to Multiple-step Classification .............................. 66  
4.3 MARP(Serengeti, DarkSkies) .......................................................... 81  
4.4 Accuracy (Serengeti, DarkSkies) of inferred label path (top 3 results) ... 81  
4.5 MARP - DarkSkies (without/with test question) ............................... 84  
4.6 MARP - GalaxyZoo (without/with test question) ............................... 84  
4.7 Accuracy of inferred label path (top-3 results) - DarkSkies (without/with test question) ................................................................. 85  
4.8 Accuracy of inferred label path (top-3 results) - GalaxyZoo (without/with test question) ................................................................. 85  
4.9 Accuracy (by path) of each algorithm ............................................. 86  
5.1 Cost overview for different workflows ............................................ 110  
6.1 Filters and the corresponding options in Snapshot Serengeti .............. 120  
6.2 Classification accuracy of two different workflow: optional filter step (old) vs. mandatory filter question step (new) ............................... 129  
6.3 Significant testing result for comparing the accuracy of different algorithms between old and new workflow, based on node level (upper) or path (lower) ...................................................... 131
Acknowledgements

I would like to take this opportunity to thank my supervisors Prof. Elena Simperl and Adriane Chapman. My research would have been impossible without the guidance, support, encouragement, and valuable feedback they have provided during my whole PhD.

I am profoundly grateful to the reviewers who have provided critical comments to my journal papers to help me improve my work. I would also like to thank my examiners and the researchers who I met in workshops, conferences and doctoral consortium, for their candid and valuable comments to my work during different stages of my PhD study.

I would like to thank my family members for their wholeheartedly support for my PhD and my life in general. I would like to express my gratitude to my friends, fellow doctoral students/colleagues for their kind support and encouragement along the way.
Chapter 1

Introduction

This chapter starts with the status quo of crowdsourcing and its existing challenges. It highlights the challenges that motivate the studies in this thesis and presents two research questions this thesis addresses. Contributions of this thesis are presented, and the structure of the thesis is outlined.

“三个臭皮匠，胜过诸葛亮。”

– Unknown (Kuan-Chung and Brewitt-Taylor, 2011)

“Three cobbler with their wits combined equal Zhuge Liang, the master mind.”

– Translation of Above

The saying derived from one of Chinese most famous novels, “Romance of Three Kingdom”, might sound ridiculous and impossible that how could three shoemakers surpass Zhuge Liang the master mind. This ancient Chinese proverb is based on stories from the Chinese three kingdom period and essentially means that “The wisdom of the masses
Chapter 1 Introduction

exceeds that of the wisest individual.”\(^1\,^3\). There is a similar expression in western saying: “Two heads are better than one”\(^4\). Nevertheless, these sayings have the same emphasis on the wisdom of the crowd and the importance of solving problems by collective intelligence.

“Crowdsourcing” (Howe, 2006) has become popular since Jeff Howe “invented” the term. Crowdsourcing essentially means that having the work that was traditionally done by experts or employees in an organisation “outsourced” to the crowd\(^5\). In the past decade, Microtask crowdsourcing has been used to help solve a wide range of problems that traditionally are expensive for experts to do or difficult for machines to handle, covering ontological engineering (Markotschi et al., 2010; Eckert et al., 2010a; Noy et al., 2013a; Sarasua et al., 2012; Mortensen et al., 2013), Linked Data management and quality assessment (Simperl et al., 2011a; Acosta et al., 2013a), Semantic annotation\(^6\), image tagging (Von Ahn and Dabbish, 2004; Liu et al., 2012), relation extraction (Aroyo and Welty, 2013a), query processing (Acosta et al., 2012), proofreading (Bernstein et al., 2010), literature evaluation (Brown and Allison, 2014a), translation (Zaidan and Callison-Burch, 2011), classification (Ho et al., 2013; Shamir et al., 2014; Lintott et al., 2011) and many citizen science (Silvertown, 1888; Cohn, 2009) projects\(^7\) such as Zooniverse\(^8\) and Stars4All\(^9\). Crowdsourcing has been applied in different ways that could help improve our daily life, such as creating and maintaining street maps\(^10\), or monitoring traffic\(^11\) in local areas (Artikis et al., 2014; Chatzimilioudis et al., 2012; Yan et al., 2009). More recently, it has also been used in various disaster management and relief activities (Zook et al., 2010; Gao et al., 2011; Salisbury et al., 2015; Ramchurn et al., 2015) to help more efficiently respond to the emergency situations. Early this year, due to the drone flying accidents near airports in the UK, the government has initiated a program to involve volunteers to help police skies over airport skies\(^12\).

While the wisdom of the crowd (Howe, 2006) has shown great potentials in above areas, it also exhibits many challenges (Kittur et al., 2013; Buettner and Buettner, 2016) in crowdsourced work either done by volunteers (Raddick et al., 2010; Lease, 2011; Newman et al., 2012) or paid crowd workers (Kittur et al., 2013; Demartini, 2015; Bernaschina et al., 2015). Researchers have looked at the challenges from different aspects (Figure 1.1), ranging from task design (task decomposition/microtasking to make it clear and easy to perform (Kittur et al., 2011; Cheng and Bernstein, 2015; Sigurdsson et al., 2016), workflow to improve the user participation and efficiency (Little et al., 2009; Bernstein

---


\(^3\) [https://www.phrases.org.uk/meanings/two-heads-are-better-than-one.html](https://www.phrases.org.uk/meanings/two-heads-are-better-than-one.html)

\(^4\) [https://www.wired.com/2006/06/crowds/](https://www.wired.com/2006/06/crowds/)


\(^7\) [https://www.zooniverse.org](https://www.zooniverse.org)

\(^8\) [http://www.stars4all.eu/](http://www.stars4all.eu/)

\(^9\) [https://www.openstreetmap.org](https://www.openstreetmap.org)


Chapter 1 Introduction

et al., 2010; Hu et al., 2010; Kulkarni et al., 2012; Haas et al., 2015; Goto et al., 2016), user interface design to facilitate the crowdsourcing process (Heer and Bostock, 2010; Bernstein et al., 2011; Sprinks et al., 2017), motivating the crowd (individually or in group) (Zheng et al., 2011; Kaufmann et al., 2011; Chandler and Kapelner, 2013; Pilz and Gewald, 2013; Mao et al., 2013; Krause and Kizilcec, 2015), to quality assurance (spam identification and prevention, dynamic task allocation (Ho et al., 2013; Difallah et al., 2013a; Karger et al., 2014), data aggregation to minimise the effect of noise and infer correct answer (Demartini et al., 2012; Kamar et al., 2012a; Hung et al., 2015; Gurari and Grauman, 2016a; Zheng et al., 2017)). These aspects are inextricably intertwined and affect the quality of the crowdsourcing result in a complex manner. As a result, existing studies normally investigate one single specific area by fixing the remaining aspects. This thesis is going to tackle two tightly related aspects of microtask crowdsourcing in the context of entity/image classification task, investigating how workflow designs and aggregation methods could help improve the quality of classification result in multiple-step classification tasks.

Figure 1.1: “A proposed framework for future crowd work processes to support complex and interdependent work.” (Kittur et al. (2013))

1.1 Motivation

Microtask crowdsourcing has attracted interest from researchers, businesses and government as a means to leverage human computation into their activities in a fast, accurate and affordable way. Over the last few years, we have seen microtask crowdsourcing
is used in a wide range of areas, as shown earlier. The underlying model is relatively straightforward: a problem is decomposed into smaller components that can be tackled independently by several people. Their individual outputs from all participants are then compared and consolidated into a final solution (Shahaf and Horvitz, 2010). However, none of these steps are actually easy: some problems are less amenable to microtasking and need to be turned into bespoke microtask workflows; the performance of the crowd varies; and determining which answers are the most useful ones can be both complex (Kittur et al., 2008; Snow et al., 2008; Vickrey et al., 2008; Bernstein et al., 2010) and computationally expensive (Demartini et al., 2012; Wiggins et al., 2011; Bachrach et al., 2012a; Venanzi et al., 2014, 2016).

Classification is the process of dividing things into groups according to their type. There are many types of classifications, document/text classification, entity classification, images/video/audio classification, etc. For instance, Named Entity Recognition and Classification (NERC) was done in the past decades mainly using automatic approaches, ranging from rule-based algorithms in the early days to later on machine learning algorithms (Nadeau, 2007). Similarly, in images classification (Lu and Weng, 2007; Kamavisdar et al., 2013), audio (Li et al., 2001; Guo and Li, 2003; Lee et al., 2009), or video classification (Zhou et al., 2000; Brezeale and Cook, 2008; Karpathy et al., 2014), various automatic techniques are employed to classify the corresponding objects. Using these machine-based mechanisms has two major challenges: accuracy of the machine algorithms is not satisfactory, and a large amount of training data are needed to improve the classifier accuracy (Zhang, 2000; Weiss and Provost, 2003). As crowdsourcing became popular in recent years, classification has become one of the most common types of tasks we have seen in various microtask crowdsourcing practices where a closed-ended question is asked, and a finite set of pre-defined categories or options is provided (Jain et al., 2017). Crowdsourcing can help with the classification process in two ways: provide labels for the objects which can be used as training data for improving machine learning algorithms (Ho et al., 2013; Costa et al., 2011; Cheng and Bernstein, 2015) or as input for statistical inference algorithms to infer correct labels (Liu et al., 2012; Willett et al., 2013); validate the classification result from automatic approaches (Yan et al., 2010).

Not all classification tasks are simple one-step classification. More often, we see relatively complicated classification tasks that need to be split into smaller relevant tasks. Taking GalaxyZoo project as an example, the steps for a user to complete a classification, the type of question and number of available options at each step, all can vary due to the fact that the relatively difficult task has been split into several smaller dependent or independent microtasks which identify characteristics of the objects being classified. In this research, we define such scenarios with multiple steps and each step has one classification question to be answered by the crowd as Multiple-Step Classification. Taking

\[\text{http://dictionary.cambridge.org/dictionary/english/classification}\]

\[\text{https://www.galaxyzoo.org/}\]
Figure 1.2: Example of a Multiple-Step Classification task from Snapshot Serengeti. The crowd is asked to choose the animal type and estimate how many animals are in the picture. Wider arrows indicate the corresponding choices are popular with the crowd.

Snapshot Serengeti project as an example, pictures were taken in the national park in Tanzania are classified on-line by thousands of volunteers. The task involves the same number of steps (classification questions) if the picture does have animals in it, with each step identifying different aspects of the animal, as shown in Figure 1.2. Further investigation is needed to understand what kind of task and workflow design can produce a better result for such Multiple-step Classification cases, and what is the suitable aggregation approach to infer true label and assess the quality of the classification in such cases.

In a typical microtask crowdsourcing classification scenario, from the requester’s point of view, there are lots of factors to consider. They can be broadly divided into two categories: common factors such as monetary cost, turnaround time, demographics and expertise of the crowd, spam detection and control, incentive mechanisms; task-type specific factors such as task design and aggregation of the corresponding collected data that are dependent upon the types of tasks (Gadiraju et al., 2014). For instance, a task of Content-Creation (CC) type, such as writing (Bernstein et al., 2010; Kittur et al., 2011) and a task of Interpretation and Analysis (IA) type, such as sentiment analysis (Cambria et al., 2010; Liu et al., 2012) demand different task design principles and require different ways to aggregate the result. This research focuses on two of these important challenges in classification tasks: task and workflow design which involves how to decompose the task into smaller units and how to chain related units in a flow to make it fairly easy to be answered, and quality assessment approaches which encompass using inference algorithm(s) to aggregate the crowdsourced data and assess the classification quality correspondingly). Task and workflow design (Little et al., 2010b; Kittur et al., 2011; Demartini et al., 2012) are crucial in ensuring the task is outsourced properly and well understood by the crowd, mitigating the chance of low-quality input (or spam in the

https://www.snapshotserengeti.org/
Chapter 1 Introduction

paid crowdsourcing context) and keeping the crowd engaged. Task and workflow design are essential to obtain a high quality and quantity of input from crowd workers. Quality assessment techniques can be used either on the fly (Ipeirotis et al., 2014) (during the task running that can be used to optimise task assignment hence reduce cost), or in the post-aggregation (Whitehill et al., 2009; Ipeirotis et al., 2010; Bachrach et al., 2012b; Difallah et al., 2015) to assess the overall quality of the classification. However, it is challenging to assess the quality of the result. First, a gold standard is expensive to create, and most of the time there are no gold answers available to evaluate the quality produced when executing different workflows. Second, existing research does not have a clear and straightforward inference approach for multiple-step classification tasks, although a few works have attempted to tackle specific instance/domain under multiple-step classification contexts (Dumais, 2000; Su et al., 2006; Parameswaran et al., 2011; Kim et al., 2002a; Wu et al., 2012; Bragg et al., 2013; Kamar and Horvitz, 2015; Otani et al., 2016). Both aspects (task and workflow design, and quality assessment) are interrelated and have not been sufficiently studied yet. Though there is some initial research in complex task and workflow (Kittur et al., 2011; Liu et al., 2012; Kittur et al., 2013; Tran-Thanh et al., 2015; Yang et al., 2016; Borromeo et al., 2016), we still have little knowledge of the broader design space and the possible approaches to aggregate multiple independent judgments more accurately for complex tasks which may have dependencies between microtasks.

1.2 Research Questions

This thesis focuses on classification tasks and investigates the following:

- **RQ1**: How to assess the quality of a multiple-step classification?

- **RQ2**: Given a fixed number of available categories, how does different task and workflow design affect the quality of the classification result? When the classification involves a large number of options, whether the multiple-steps design can achieve a higher classification accuracy?

The research is intended to answer the above questions by looking at two classical classification areas, entity classification and image classification, to fully understand the challenges and explore the factors that may help improve the quality of classification result. DBpedia acts as the source for entity classification, while Zooniverse\textsuperscript{8} and Stars4all\textsuperscript{9} provide datasets for exploring image classification.

To address **RQ1**, multiple-step classification cases are modelled as a graph and aggregation strategy will be investigated based on existing established inference algorithms. Investigation on **RQ1** gives us the proxy to assess quality in multiple-step classification
as a lot of tasks are not simple one-step tasks and inferring the correct labels for tasks
with such complex workflow has not been fully studied yet. Three different image clas-
sification tasks from popular citizen science projects will be used during the process to
test the aggregation model.

To address RQ2, we first explore different workflows and evaluate the quality of clas-
sifications resulting from those workflows. Investigation on the first part of RQ2 will
mainly use DBpedia entity classification as an instance to test the proposed classifi-
cation model comprised of predictor, error detecting and error correction process. For
answering the second part of RQ2, drawn from the knowledge on classification workflow
alternatives, multiple-step classification, and its aggregation, we could validate the effect
of multiple-step design in a controlled environment. One multiple-step classification task
is chosen, in which one of its classification involves a large number of options that makes
it possible to be turned into a further multiple-step classification. This test will confirm
whether multiple-step classification can consistently achieve high-quality results.

1.3 Contributions

This thesis focuses on microtask crowdsourcing and hybrid human-machine classification
with more than one classification step. The study is influenced by existing classification
theories (Rosch and Lloyd, 1978; Wisniewski and Medin, 1992; Batley, 2014). Alterna-
tive workflows have been explored to allow further study of the problem space. Exper-
iments were carried out with representative existing classification tasks to investigate
how task and workflow design, and aggregation method can impact the quality of clas-
sification. As a result, the study can produce useful insight into effective classification
task design, particularly for the tasks that involve many classification categories. Major
contributions include:

• Conceptualisation of multiple-step classification problem as a path searching prob-
lem in a graph, and investigation of aggregation in such multiple-step classification
scenarios. (related to RQ1)

• Investigated different multiple-step image classification tasks from popular citizen
science projects: classification quality is algorithmically assessed. Analysis pro-
vides useful insights on both multiple-step task design and quality assessment.
(related to RQ1, see our publication Bu et al. (2018))

• Proposed alternative workflows for classification and presented insights derived
from using entity classification task as a case study. (related to RQ2, see our
publication Bu et al. (2019))

• Comparison of the different design on the same set of image classification to val-
idate mandatory classification questions produce higher accuracy than optional
filters. Design recommendations for achieving quality results in a classification task are discussed to help task requesters choose the proper strategy. (related to RQ1 and RQ2)

During the process, two main models were studied: one for the classification workflow, the other is for aggregation in multiple-step classification (Figure 1.3).

- The classification workflow model presents a pipeline approach where three general classification workflows can be utilised for most of the classification tasks.
- The aggregation model mainly investigates the methods to infer the most likely correct answer for a multiple-step classification task: alternative aggregation approaches are proposed to help assess quality, and interesting insights are obtained from the observation on how a certain answer inference algorithm improves accuracy for a specific type of multiple-step classification.

The overall output of the research was concluded in a list of design guidelines for a classification task to achieve quality results.

1.4 Structure

The rest of this thesis is structured as follows: it starts with the background in Chapter 2 to present all the related concepts and works that are foundations to understand this
work. Chapter 3 presents the overall methodology in addressing the aforementioned research questions. Two components are proposed allowing testing the performance of different workflows and identifying aggregation strategy. Experiments to evaluate these two components could help identify some of the important factors that affect the classification quality which then can be validated in a controlled environment, and eventually used to guide the classification task design and the answer inference process. Chapter 4 introduces the conceptualised graph model for multiple-step classification, applies the dependency-aware aggregation model to infer the correct path in multiple real-world classification datasets and assess the corresponding classification quality. Chapter 5 is dedicated to the classification workflow model and the corresponding experiments to evaluate alternative workflows in an entity classification scenario. In chapter 6, the study extends on the insights obtained from Chapter 4 and 5, investigating the impact of multiple-step user-driven filter selection (compulsory step), comparing with the baseline case where filters are optional to use during classification. Finally, Chapter 7 concludes our research in multiple-step classifications, followed by an overall discussion on the limitations of this work and potential future work.
Chapter 2

Background

This chapter is structured to introduce the background of the areas related to my research. It starts with microtask crowdsourcing and its different dimensions in section 2.1. It then introduces how classification is currently done, manual, automated, and hybrid approaches in section 2.2. Existing research on task and workflow design from the classification task perspective is reviewed in section 2.3. Section 2.4 looks at quality dimensions, metrics and the methods that can be utilised to infer the correct label and assess the quality of crowdsourcing classification results.

2.1 Microtask Crowdsourcing

This section reviews different aspects of crowdsourcing and relative interesting features that could determine an effective task. From there, it focuses on the most relevant areas to this research: task and workflow design (2.1.2), and quality assessment (2.1.3) in the classification task context. Other key aspects relevant to the quality result of the crowdsourcing are acknowledged in section 2.1.4, including the crowd (motivations/incentives) and corresponding quality assurance mechanisms (consisting of task assignment strategies).

In recent years, we have witnessed a trend of leveraging the wisdom of the crowd (Howe, 2006; Kittur et al., 2008) to help tackle the problems and tasks that are either extremely
time-consuming and expensive for domain experts to complete or difficult for the machine to handle automatically (Quinn and Bederson (2011)). Microtask crowdsourcing, by outsourcing the work to the crowd in smaller parts, has been widely applied and studied in various areas. In microtask crowdsourcing, a task is usually broken down into smaller “microtasks” so that it can be solved by non-experts, normally by many crowd workers in parallel in a shorter time with lower cost, and is expected to have equivalent or better quality than by an expert or a small group of expert users. To fully leverage the wisdom of the crowd and achieve the goal of completing tasks in a fast, cheap manner with good quality, one should look at the important crowdsourcing dimensions (Malone et al., 2010) – what? who? why? and how?

2.1.1 Basics and Characteristics of Microtask Crowdsourcing

**What:** It represents the goal of the task, either can be “Create” or “Decide”, according to Malone et al. (2010). Tasks of the “Create” type require generating something new, such as producing a new article, designing a product logo, drawing a map, writing software code or collecting data (e.g. taking picture of a bumble bee and uploading it\(^1\)), etc. “Decide” tasks are mostly evaluating or selecting an alternative based on given contents, such as transcription of audio or digitalised archives, sentiment analysis of tweets or news articles, evaluation of the relevance of search results, classification of entities, images classification, etc. (Gadiraju et al., 2014, 2015a) analyse hundreds of tasks deployed on Figure Eight\(^2\)(originally known as Crowdflower\(^3\)) and Amazon’s Mechanical Turk\(^4\), and present six goal-oriented categories to classify the microtasks, including information finding (IF), interpretation and analysis (IA), content creation (CC), content access (CA), survey (S), verification and validation (VV). In most cases, a task to be outsourced to the crowd may involve more than one type of aforementioned microtasks. For instance, to classify an entity, crowd workers may be asked to access a paragraph that describes the entity (CA), then classify it to a specific category based on a given list of categories (IA) or validate whether it belongs to a given category (VV). The “What” dimension then largely affects other dimensions of the crowdsourcing practice in terms of who will participate, why they actively participate, and how they can participate and carry out the tasks.

**Who:** This dimension is mainly the participants (often referred to as crowd workers) who will undertake the crowdsourcing tasks. Crowd users normally use an online platform to undertake the activities requested by the task requester, whether they access an online website directly or use their mobile device to collect and submit data. Theoretically, anyone can participate in the microtask crowdsourcing activities. In reality, there

\(^{1}\)http://www.bumblebeewatch.org/  
\(^{2}\)https://www.figure-eight.com/  
\(^{3}\)https://www.figure-eight.com/focused-future-new-name/  
\(^{4}\)https://www.mturk.com/mturk/welcome
are two major types of platforms each of which has very different strategies in how they recruit and profile crowd workers, and hence set apart the crowd workers into paid workers and unpaid workers. Platforms like Mechanical Turk\(^4\), Figure Eight\(^2\), Clickworker\(^5\) or MicroWorker\(^6\) are commercialised platforms that only allow registered users to participate. On these platforms, requesters deploy their tasks and pay the crowd workers to do the tasks in order to have the job done faster, cheaper and with good quality. These goals impose the requirement to profile crowd workers so that better control can be done in terms of crowd worker selection based on previous performance. Such platforms keep a record of each crowd worker’s profile, from demographic information to performance related information, which makes it feasible for the requesters to distribute their tasks to a specific type of crowd (Difallah et al. (2013b)). Other platforms are basically open to everyone who is willing to contribute without being paid and normally does not require registration. The goal of those platforms and the tasks run on them are non-profit and are mainly for the public benefit or scientific research. For instance, citizen science platform Zooniverse\(^8\) and crowdsourcing platform CrowdCrafting\(^7\) allow anonymous users to contribute without having to register with the system, only prompting the user to register after a certain number of tasks have been done. Unpaid crowd workers include those who explicitly volunteer to undertake the crowdsourcing tasks, like in most of the citizen science projects\(^8\), as well as those who might be unaware of their participation of crowdsourcing activities, such as reCAPTCHA\(^9\) by Von Ahn et al. (2008) and games that with a purpose (GWAP) that actually solve a wide range of problems including image annotation, image search, natural language processing, ontology alignment and even scientific research to produce protein structure, RNA molecule, and so on (Von Ahn and Dabbish, 2004; Von Ahn, 2006; Vickrey et al., 2008; Bernstein et al., 2009; Thaler et al., 2011; SemanticGames, 2012; Simperl et al., 2013b; 510, 2014). In addition to the crowd workers that are not experts, De Boer et al. (2012) brought up the idea of “nichesourcing” using a small group of expert crowd to do the work, which has also been applied in a few cases, such as cultural heritage project by Oosterman et al. (2014) and specific linguistic research\(^10\). Using paid or unpaid crowd workers, expert or non-expert to do the task to some degree will affect how the requester chooses incentive mechanisms to encourage participation and good quality of work, as well as how they will design the task. The why dimension explains further on the motivations of the crowd workers and the how dimension will elaborate on task and workflow design.

**Why:** This dimension is related to the persons who will undertake the crowdsourcing activities and their motivations. Malone et al. (2010) suggests three high-level categories that motivate people to participate: money, love, and glory. Malone et al. (2010) points

---

\(^4\)http://www.clickworker.com/
\(^5\)http://www.microworker.com/
\(^6\)http://www.crowdcrafting.org/
\(^7\)https://en.wikipedia.org/wiki/List_of_citizen_science_projects
\(^8\)https://www.google.com/recaptcha/intro/
\(^9\)http://doria32-kk.lib.helsinki.fi/handle/10024/111870

---
out two basic rules that could motivate participants and improve crowdsourcing efficiency – using love or glory (or both) incentives may reduce cost; using money and glory can help reduce the time. Howe (2008) has examined the motivations that would engage the crowd into certain activities and point out monetary reward is not necessarily the key motivator for people to voluntarily participate. The study from Borst (2010) suggests intrinsic motivations are the key drivers for participation while extrinsic motivations such as money could play a negative role. Shadbolt et al. (2013) echos this result and states that people are intrinsically motivated when they are learning new knowledge, sharing their knowledge, contributing to the community, or feel a sense of belonging to the professional group. Another intrinsic motivator worth mentioning is the fun elements which have been explored since a decade ago, including various gamification mechanics (Zichermann and Cunningham, 2011; Deterding et al., 2011). Gamified design has also been applied in real crowdsourcing applications11 and scientific projects such as FoldIt12 and EyeWire13. Fun elements may be good in attracting crowd worker’s interest at first place, then other incentives, whether it is virtual rewards such as badges, leaderboards, bonus points, or monetary reward, or simply the activity itself providing a good opportunity for people to learn or share new knowledge, will pretty much depend on the type of task. It will be easier to control the incentives once the “how” aspects have been designed (Simperl, 2014).

How: This “how” dimension represents how the crowdsourcing task will be carried out. In a broad sense, a task can be carried out either independently or collaboratively by the crowd, or even as a competition in some cases. Malone et al. (2010) correspondingly characterises these different ways of doing crowdsourcing into collection, collaboration, and contest. In a specific sense, “how” is closely related with “what”, “who” and “why” aspects, and “How” will accordingly decide what motivation and rewards mechanisms to be used, how the jobs are designed and distributed, how quality control will be done and what type of aggregation will be used and when to use it. These are comprehensively reflected in the crowdsourcing foci by Kittur et al. (2013): platform, motivation, hierarchy, reputation, task decomposition, job design, task assignment, collaboration, real-time work, workflow, quality assurance, and crowds-AI interaction (1.1).

To run a crowdsourcing task, a platform is required to manage the tasks and crowd workers. The features a platform can provide will influence on other aspects of the crowdsourcing, such as recruiting workers, the interactions between task requesters and crowd workers, interactions between workers, how motivation can be implemented, the way tasks are assigned, what quality control mechanism can be used and how data can be aggregated, etc. To recruit and maintain crowd workers, the task requester needs to incentivise the workers and implement the corresponding motivation and rewards that

---

11http://badgeville.com/customer/case-study/emc
12http://fold.it
13http://eyewire.org
are beyond monetary rewards (Kittur et al., 2008; Rogstadius et al., 2011), and incorporate reputation system in the platform to better support both crowd workers and task requester. The crowd workers also need to be properly selected and organised to carry out the task effectively (Acosta et al., 2013b; Sigurdsson et al., 2016; Barbosa and Chen, 2019). Existing platforms do not seem to offer sufficient and flexible support on worker selection process (Kittur et al., 2008; Difallah et al., 2013a; Satzger et al., 2013), most of the platforms only offer simple worker profiling by categorising workers into different levels such as in Figure Eight ², which does not make it easy for the requests to identify workers who are suitable for the type of tasks they are crowdsourcing. The support of allowing task requester to run a qualification test does not exist (e.g. Crowdcrafting⁷) or is not flexible (e.g. Figure Eight²). In addition, other quality control mechanisms need to be considered to ensure the data collected during the task execution has good quality or low-quality input is filtered out afterwards. Though quality control has been a major concern of crowdsourcing work and lots of research has been conducted in the past (e.g. through redundancy (Callison-Burch and Chris, 2009), iterative tasks (Little et al., 2009), validation by expert (Khattak and Salleb-Aouissi, 2011), using Machine Learning Strategies or statistical algorithms to detect spammers or block low-performance workers (Lease, 2011; Ipeirotis et al., 2010), probabilistic method to estimate the true label from noisy labels (Raykar et al., 2010; Whitehill et al., 2009)), it remains a challenge (Gadiraju et al., 2015a,b).

In order to be answerable by a wide range of workers, the complex task is decomposed into smaller sub-tasks making them manageable and easy to handle by the crowd. The way task is decomposed is important (Shahaf and Horvitz, 2010; Lease, 2011) in minimising the required cognitive load for the crowd workers to complete a task, as well as making the sub-task clear, attracting, fulfilling to the workers and fault-tolerant through proper job design techniques. The decomposed sub-tasks are independent or interdependent and need to be connected via effective workflows (Bernstein et al., 2010; Kittur et al., 2011; Tran-Thanh et al., 2015) and properly implemented on a platform where dependencies between sub-tasks are well supported (for instance, CrowdCrafting⁷ only supports sequential sub-tasks one after another instead of the flexibility of only presenting certain sub-tasks based on the output of the previous sub-task). The tasks are organised via multiple-step workflow, and then need to be distributed to the crowd workers to complete offline or in a real-time fashion, collaboratively (Rokicki et al., 2015; Feyisetan and Simperl, 2016) or individually (Feyisetan et al., 2015a), synchronously or asynchronously, based on a “first-come, first serve” task assignment strategies or other advanced strategies such as costs-oriented (time and/or monetary cost) and quality-oriented approach. For instance, ZenCrowd (Demartini et al., 2012) uses dynamic assignment based on user performance. Ho et al. (2013) presents an adaptive classification task assignment approach. (Gurari and Grauman, 2016b) proposes to use the answer diversity to decide whether to solicit more answers from the crowd. Last but not least, corresponding to the “Crowds Guiding AIs” and “AIs Guiding Crowds” by (Kittur et al.,
the interaction between human and machine could be utilised to make the hybrid human-computer system more efficient. “Crowds Guiding AIs” focus on the aspect of how crowd input can help train and improve automated machine approach, while “AIs Guiding Crowds” refers to how automated machine approach can help assure or improve the input quality from the crowds. Such interaction should be part of the crowdsourcing platform and an integral part of the whole crowdsourcing process. In particular, there are potential needs to move from the simple automatic way of controlling a workflow or task assignment to a “richer, mixed-initiative settings where crowds and AIs jointly teach each other, and jointly control the work process”.

Each of these aforementioned foci has identified different challenges for future research. In general, it can be summarized into person-centric aspects and process-centric aspects. This study mainly explores the process-centric aspects with a focus on task and workflow design, as well as the quality assessment aspects.

2.1.2 Task and Workflow Design

In crowdsourcing, a problem needs to be decomposed into smaller, fine-grained microtasks and then arranged in a workflow for more effective processing. In general, a workflow consists of a set of microtasks; the microtasks are sometimes of different types and can be dependent or independent of each other. For instance, the find-fix-verify workflow proposed by Bernstein et al. (2010) uses microtask crowdsourcing to proofread and shorten text in three steps: finding areas of improvement in the text; fixing or improving them; and verifying the quality of the changes. In each step, the crowd is asked to carry out the same type of microtask, sometimes iteratively. In (Kittur et al., 2008, 2013; Acosta et al., 2013a), researchers have proposed to group the same or similar microtasks into batches to facilitate learning effects. Previous studies have also shown that task performance can be improved as a function of several factors, such as the design of tasks and workflows (Kittur et al., 2008; Bernstein et al., 2010; Demartini et al., 2012; Wiggins et al., 2011), motivation and incentives (Zhang and Zhu, 2011; Mao et al., 2013; Ramchurn et al., 2013), and training (Dontcheva et al., 2014; Staffelbach et al., 2015). This section focuses on task and workflow design studies, some of which have already considered motivation/incentives/training.

Task and Workflow Design is normally specific to Task-type: “Create” Type and “Decide” (Malone et al., 2010) might require different task design so that high enough quality data can be collected accordingly. For instance, a logo design task and a sentiment analysis task would have different UI designs to accept user input, different mechanisms to “control” quality data is collected, etc. However, sometimes certain design principles are applicable to both types. Aforementioned work by Bernstein et al. (2010) provides a typical workflow later has been adopted by other tasks of “Create” type, as well as “Decide” type. For a similar writing task, Kittur et al. (2011) proposes the CrowdForge
framework and evaluate it upon an article writing task (“Create” Type), which shows that it could produce higher quality articles through breaking down the task and automatically managing the flow. CrowdForge follows the idea of MapReduce\(^{14}\) loosely and introduce the partition, map and reduce task in a simplified distributed computing fashion to allow breaking down to sub-tasks, distributing to more than one workers, and automatically aggregating answers from multiple workers. Instead of focusing on the writing task itself, Kaur et al. (2018) uses the help from the crowd to create vocabulary-based action plans for the collaborative writing task. Informed by the way Wikipedia\(^{15}\) works, Little et al. (2009) investigates an iterative paradigm and how it can be applied to different tasks, including image description, handwriting recognition, etc. A similar iterative idea is used in a Decide type speech-to-text task (Li et al., 2011). Decide type task such as audio file transcription may involve distinct design aspects, but sometimes could use similar workflows investigated on Create type tasks: Li et al. (2011) presents a transcription process following the MapReduce principle\(^{14}\), in which an audio is divided into ten second short clips and re-join transcripts at a later stage, in order to handle the challenge of quality control in transcription task. Sigurdsson et al. (2016) especially focuses on temporal data, video in their study, and show that asking many binary questions all at once might be the most cost-effective way of receiving high-quality annotations, without breaking down the video into shorter clips. Abad et al. (2013) shows that use of live feedback in the speech transcription task could help improve the quality. Within this work, Classification task we focus on is the “Decide” type where classes/categories are provided for the user to choose from. The corresponding microtask design is reviewed in section 2.3.

There are other studies that focus especially on Task User Interface to make it interesting, easy to understand and easy to use. Finnerty and Kucherbaev (2013) demonstrates that simpler and less demanding UI design can lead to more accurate results in a classification task with six categories. Feyisetan et al. (2015b) investigates image labeling and text annotation, using the game-based interface of WordSmith and show that game-based interface has better user engagement, higher quantity and quality result compared to the traditional crowdsourcing platform\(^2\). Von Ahn et al. (2008) presents a brilliant idea to leverage the human linguistic ability to help digitise the archive/books, while the task is presented as part of the process to distinguish human and bot, for a person to complete (without realising it is a “transcribing text from image” task). Hahn et al. (2019) looks at how to embed microtask (part of a writing task) casually into Facebook and shows the task can be completed with high quality, as well as helping the user concentrate more on their own work. Heer and Bostock (2010) carries out graphical perception experiments on Mechanical Turk\(^4\) and explored good design guidelines such as contrast adjustment of chart gridlines, chart size and gridline spacing. Bernstein et al. (2011) especially focuses on the design of real-time crowd-powered interfaces

\(^{14}\)https://www.ibm.com/analytics/hadoop/mapreduce

\(^{15}\)https://en.wikipedia.org/wiki/Main_Page
and explored the interactive UI that not only allows the crowd to select a photo, using a deformation tool to produce a picture based on the given description, but also the requester to provide feedback.

Facilitating parts of a task, such as instructions (Gadiraju et al., 2017; Gillier et al., 2018; Bragg and Weld, 2018; K. Chaithanya Manam et al., 2019), examples, help or tutorial (Dontcheva et al., 2014; Staffelbach et al., 2015; Liu et al., 2016), feedback (Lampe and Johnston, 2005; Stumpf et al., 2008; Dow and Klemmer, 2011), and forum (Newman et al., 2010; Luczak-Rösch et al., 2014), is also vital to the success of a crowdsourcing task. Gadiraju et al. (2017) conduct a study of various tasks using five years’ datasets from Mechanical Turk\(^4\), investigating task clarity (description and instructions) that can be represented by a set of features, and propose a method to help predict the task clarity based on the goal and role in the task instructions. Gillier et al. (2018) investigates the effect of different type of task instructions (Unbounded, Suggestive, Prohibitive) on creative tasks, and confirm that task instruction plays an important role: for instance, a task with suggestive instructions (the crowd is asked to produce ideas that can improve existing product) will lead to lower originality; while unbounded task instruction could promote more original and feasible ideas. Bragg and Weld (2018) presents SPROUT to help the task requester write instructions by getting feedback from the crowd to understand the areas need to be improved/clarified and enhance the task instructions accordingly. In the recent work by (K. Chaithanya Manam et al., 2019), a system called TaskMate is proposed allowing crowd workers to help refine the task instruction with minimal efforts from the task requester. Dontcheva et al. (2014) demonstrates that when interactive step-by-step tutorials are provided, the crowd can produce high-quality results for image editing task using “LevelUp for Photoshop” (the platform the authors built to support their study). Staffelbach et al. (2015) looks into a complex task which is not able to be split into an easy-to-do smaller tasks and show that if a comprehensive tutorial is provided, the crowd could produce a comparable result to the experts who have prior knowledge of related task areas. Liu et al. (2016) presents “Gated Instruction” (GI) which provides an interactive tutorial and gives feedback during the crowd’s learning/training stage. It shows that it is able to create more accurate annotations for relation extraction tasks. Forums and online communities are also very helpful in engaging the crowd in crowdsourcing tasks: Newman et al. (2010) investigates web mapping application in the citizen science context and discover that volunteers have the need to communicate with each other. Luczak-Rösch et al. (2014) analyses ten citizen science projects from Zooniverse\(^8\) and show that the online community can engage the volunteers to contribute on more projects.

As demonstrated from above studies, considering the complexity (Yang et al., 2016) of the task, expertise level of the crowd, the ways to formulate the task, as well as how to engage more participation, crowdsourcing tasks need to be designed in a way that makes it intuitive and easy to be understood, easy to finish and resistant to spammers.
or low-quality work. In order to make the task easy to be solved by the crowd, complex tasks are usually decomposed into smaller microtasks and workflows (Little et al., 2010a; Kittur et al., 2011; Liem et al., 2011; Lin et al., 2012; Noy et al., 2013b; Tran-Thanh et al., 2015; Fang et al., 2014) are applied to put together the subtasks and carry out the complicated crowdsourcing task in an effective manner. This is one of the important areas this research on crowdsourcing classification task covers and the related work is reviewed in section 2.3.

### 2.1.3 Quality in Microtask Crowdsourcing

Quality Assessment in microtask crowdsourcing refers to the evaluation of quality of the workers’ work and is part of the bigger concept of quality management (Ipeirotis et al., 2010; Gelas et al., 2011; Yu et al., 2012; Allahbakhsh et al., 2012), quality assurance/control (Matthew Lease, 2011; Daniel et al., 2018). Quality assurance essentially involves the process of evaluating the quality of the work based on a certain standard, and take action to help control the quality, such as preventing further malicious or low-quality data is collected. Quality management involves both evaluation of workers’ performance and the corresponding management in a crowdsourcing platform (Yu et al., 2012; Allahbakhsh et al., 2012). The following paragraphs first focuses on the evaluation/assessment related aspects, which are used in the control/management process which are reviewed afterwards.

First, quality can be assessed based on different criteria (2.4.2), as it has many dimensions (2.4.1). Under the crowdsourcing context, it depends on the type of the data, which is decided by the task type (Malone et al., 2010; Gadiraju et al., 2014, 2015a). Obviously, depending on whether it is a concrete answer (e.g. category), a number or an approximate range, or subjective input (e.g. proofreading, reviews, writings such as paragraphs for Wikipedia articles), quality could refer to different dimensions accordingly. As the “What” aspect in Basics and Characteristics of Microtask Crowdsourcing (section 2.1.1) indicated, “Decide” and “Create” task require diverse quality measures. In a “Decide” task such as transcription of audio recordings by Gelas et al. (2011), quality is assessed by calculating the accuracy with available gold standards. Liem et al. (2011) also investigate the audio transcription task with a focus on the workflow design and evaluate the quality based on similarity score of the transcripts produced from two different pathways, when the gold standard is not available. Meanwhile, Abad et al. (2013) choose to use both the Word Error Rate (WER) and ROVER Accuracy (Fiscus (1997)) to assess the quality of the transcriptions collected from crowd workers. Burrows et al. (2013) investigate the paraphrase task and use ten similarity metrics to evaluate the answers from the crowd. Sigurdsson et al. (2016) investigates temporal data annotation with a large number of target concepts to discover from the video clip, and employ a strategy of asking as many binary questions as possible, which as they put “This naturally leads
to lower recall $r$ than if we ask only a handful of questions that the workers would be more likely to read carefully” and evaluate the recall and precision accordingly. For a “Create” task, such as the task presented in Bernstein et al. (2010), a writing process has been split into “Shortn” (text shortening), “Crowdproof” (spelling and grammar checking) and “Human Macro” (interface for the crowd to help create natural language task which will be executed by crowd workers), which is then evaluated with different metrics: Shortn is quantitatively assessed to see how much (percentage) the original texts have been shortened, and how long the work time for shortening the texts; Crowdproof is evaluated by the number of errors caught and fixed, as well as the time spent; Human Macro evaluation is done using “intention” and “accuracy” to measure whether the crowd does understand the prompt provided and make a good effort on the task, and how accurate the corresponding result is. Dow and Klemmer (2011) develops Shepherd which provide the requester the ability to manage and provide feedback to the crowd, and evaluate the quality of with vs. without feedback condition in an open-ended task which asks the crowd to design text-based web advertisements, by incorporating ratings (20 points based) from independent raters (could be other crowd workers who are doing the same task in an overlap time frame). Barowy et al. (2017) particularly focuses on estimations (numbers) collected from the crowd and show VOXPL can help automatically aggregating the result with the specified confidence level, confidence interval and the given budget. In section 2.4, details of different quality dimensions, metrics and the ones commonly used in classification evaluation, will be examined in order to pick the quality metrics to be used in this study.

Quality assessment can be done with or without the gold standard. Ideally, if the gold standard is available, assessment can be done quickly by comparing the answers collected from crowd workers to the gold standard. In many cases, the gold standard is not available which is also the reason there are lots of crowdsourcing tasks seeking to get the answer or solution by gathering the crowd’s opinions. In this context, aggregation method(s) are used to infer/predict/summarise a final answer (either a concrete category, or approximate number, or collective-writing pieces, etc), which vary based on the type of task and the type of solution corresponding task seeks. Majority voting (MV) has been commonly used in crowdsourcing platforms and many microtask projects (Hung et al., 2015; Liu et al., 2012) due to its simplicity and short computing time. However, majority voting does not perform well in the domain where the crowd workers have little knowledge or objects are difficult to be classified, in terms of determining a reasonable threshold value to be confident of the aggregated result. Researchers have also tried different varieties of the majority voting algorithms, such as improved majority voting (Yang et al., 2015), weighted majority voting (Tran-Thanh et al., 2013; Zhang et al., 2017) and de-biased majority voting (Lintott et al., 2011; Willett et al., 2013). Meanwhile, researchers have proposed so-called inference algorithms, mathematical models that can automatically infer the correct solution to a given problem from a solution space defined by the crowd. For example, Ipeirotis et al. (2010) presents an algorithm
that assesses the performance of crowd workers and exploits this information to estimate
the quality of answers on Mechanical Turk. Karger et al. (2011) proposes to use message
passing to determine correct answers. Bachrach et al. (2012b) uses a Bayesian graphical
model to grade test answers in scenarios where ground truth cannot be made available.
Whitehill et al. (2009) follows an expectation maximization approach to identify correct
classifications, depending on the expertise of the workers and the level of difficulty of
the task. In the citizen science project Galaxy Zoo Supernovae, crowd answers were
analysed using a Bayesian generalisation of the same expectation maximization idea
(Simpson et al., 2011). More recently, Difallah et al. (2015) compiles a set of features
that can be used to predict answer quality, based on an analysis of Mechanical Turk
logs. Section 2.4.3 looks at the inference algorithms that are suitable for classification
label aggregation in detail.

Additionally, quality assessment can also be done on-the-fly when the task is executed,
instead of after all the tasks have been completed. On-the-fly quality assessment (De-
martini et al., 2012; Raykar, 2012; Blanco et al., 2013; Haas et al., 2015) is mainly used
as part of quality control to optimise the task assignment. Rather than allocate the
tasks on a first come first serve order as in most cases, Demartini et al. (2012) dynam-
ically assigns tasks based on user performance. Similarly, Ho et al. (2013) investigates
adaptive task assignment with a classification task and find it could produce more ac-
curate prediction with a lower cost. Gurari and Grauman (2016b) devises a system
that leverages answer diversity to predict if a crowd will agree, then decide whether
to solicit more answers from the crowd accordingly. Dynamic task assignment and the
corresponding quality assessment during task execution will not be part of this study.
This research concentrates on the classification task and workflow design and explores
how the collected data can be effectively aggregated to archive good quality results, by
comparing to the gold standards we created.

At the same time, relevant quality assurance mechanisms should be considered based on
the understanding of the crowd (and their behavior pattern), including the commonly
used approach of dynamically assigning tasks to crowd workers who are good at the
corresponding work (Demartini et al., 2012; Ho et al., 2013), or a mechanism to prevent
malicious/low-quality workers from further involvement in the task (Gadiraju et al.,
2015b). Daniel et al. (2018) states “assuring quality is, putting into place measures that
help achieve/control quality”. Achieving quality results from crowdsourcing relies on
proper task/workflow design as well as quality control mechanisms: Task and workflow
design (Bernaschina et al., 2015; Yang et al., 2016; Borromeo et al., 2016) can properly
reduce ambiguity, provide educating examples, useful hints or instructions, or frequent
feedback. Some workflow can have the input from crowd users validated by other peers
(Bernstein et al., 2010; Little et al., 2009). The crowd answers can be evaluated algo-
rithmically on the fly or after all the data has been collected to eliminate the low-quality
work or reduce its weight on final inferred result (Lease, 2011; Demartini et al., 2012;
Quality has been studied by many researchers in the past and the current research tends to separate the quality control mechanisms (Lease, 2011; Gadiraju et al., 2015a,b; Little et al., 2009) and quality assessment methods (Ipeirotis et al. (2010); Khattak and Salleb-Aouissi (2011); Raykar et al. (2010); Whitehill et al. (2009)). Section 2.4 illustrate quality assessment related aspects, including quality dimensions, measures and related inference algorithms that this research will investigate. Here let’s focus on the quality control mechanisms:

- **Before Task Run**: One of the common techniques used to control quality is to recruit the workers who are good at the given task. This is normally achieved by a qualification test where any workers can participate but only the workers who reach a certain accuracy level will be allowed to work on the task. The pre-screening method using pilot questions to filter workers has been widely used in many studies (Heer and Bostock, 2010; Thaler et al., 2012; Alonso et al., 2008). Heer and Bostock (2010) use qualification tasks to make sure the user understands the task instructions. Thaler et al. (2012) uses qualification tests to select workers whose accuracy is above 90% to participate in tasks that involving deciding whether an entity or class exist in a Wikipedia article/page and then classifying it into classes in the Proton ontology. Alonso et al. (2008) creates qualification tests to make sure the workers have certain knowledge about geography as the task is related to Information Retrieval for articles about different countries. On the other hand, there is research showing the demographics and characteristics of the workers can be used as indicators of their quality (Kazai et al., 2012). Downs et al. (2010) investigates workers’ behavior using screening questions and show that people of certain occupations tend to take the task more seriously, and men over 30 or women of any age tend to treat the tasks seriously. Crowdsourcing platform such as Mechanical Turk or Figure Eight provide basic configurations allowing the task requester to choose workers with certain traits (eg, level 1-3 for the “Contributors” setting in Figure Eight to select workers based on experience/accuracy; “country” and “language” options are also available when specifying the Contributors’ setting). Acosta et al. (2013b) only recruits workers whose previous acceptance rate is higher than 50%. Li et al. (2014) develops a crowd targeting framework which can automatically discover worker’s attributes for a task and target the corresponding task only to the workers who can potentially produce high-quality results. Gadiraju et al. (2018) show that previous behavior trace (mouse movement, key press, etc) could be leveraged to automatically classify workers into fine-grained typology so that pre-selecting specific types of workers for a given task is feasible. Barbosa and Chen (2019) focuses on enabling the task requester to recruit workers matching their specified demographic-based criteria. Sigurdsson et al. (2016) only recruits workers from the United States for their video annotation task. Recent
work from Görtz et al. (2019) develops an open source attentiveWeb and explored using an attention test to select crowd workers.

- **During Task Run**: There are also approaches for quality assurance during the task is executed. One way to constantly control quality is similar to the ones used in qualification testing, by checking the worker’s performance on-the-fly based on the available ground truth (Blanco et al., 2013) or the “likely” correct answer (Demartini et al., 2012; Talukdar and Cohen, 2012; Raykar, 2012; Ho et al., 2013). The main purpose of evaluating worker performance on-the-fly is mainly to adjust the task assignment, by removing the low-performance workers, and/or directing the task to high-performance workers. Blanco et al. (2013) mixes two Human Intelligence Tasks (HITs)\(^{17}\) with ground truth in every twelve HITs to monitor the work performance and detect workers answering randomly, so that such low-performance workers could be removed from the task and released tasks are available to qualified workers. In case a gold standard is not available, other approaches such as the most straightforward majority agreement (Snow et al., 2008; Raykar et al., 2010) or a more sophisticated approach incorporating additional statistics (Wu et al., 2012) will be used. Vuuren and De Vries (2012) compares worker’s judgements (answers) with other workers to detect random spammer (“workers that purposely randomize their responses”) and uniform spammer (“choose one label or a simple pattern to uniformly spam”). Raykar (2012) proposes SpEM, a Bayesian algorithm to iteratively estimate consensus and remove low-performance workers. The Argonaut framework presents a model for predicting worker quality based on both error score and work speed to help select trusted workers for their review work in a structured data extraction task (Haas et al., 2015). A couple of research focusing on dynamic task assignment has shown it can help improve the overall quality, such as dynamically assigning based on user performance in ZenCrowd (Demartini et al., 2012) or adaptive classification task assignment presented in Ho et al. (2013). Kazai et al. (2011) defines a typology of workers, including diligent workers, competent workers, incompetent workers, sloppy workers, and spammers. In the similar line of work, there are other mechanisms to detect “potential” malicious workers without having to aggregate the collected data yet based on specific characteristics of spammers/sloppy workers, such as spending an extremely short time on a task (Kittur et al., 2008) or spending longer than the time required (Venanzi et al., 2016). Crowdsourcing platform such as Figure Eight\(^{2}\) provide similar quality control settings for the task requester so they can set the minimum required time for a task page, as well as other useful rule-based settings.\(^{18}\) The other popular approach is focusing on using workflows to minimise the chance of low-quality work, such as the workflow designs discussed in section 2.1.2, e.g. find-fix-verify (Bernstein et al., 2010), iterative method (Little et al., 2009), or live

---

\(^{17}\)https://blog.mturk.com/understanding-hit-states-d0bc9806c0ee
\(^{18}\)https://secure.sky.com/cancel-broadband/broadband/reasons
feedback (Abad et al., 2013). Furthermore, in addition to crowd workers providing random or low-quality input, they might also use automatic scripts to “cheat” the system such as clicking all the links on the task page. Thus, it is common to use Captchas (Blum and Langford (2004)) somewhere in the task to make sure it gets the user’s attention and validate it is the human doing the task (Alonso et al. (2014)).

- **After Task Run**: Sometimes, quality control to eliminate the noisy labels/annotations is done after all data have been collected. Most of the crowdsourcing platforms allow a requester to review\(^{19}\) submitted work and reject\(^{20}\) work from sloppy workers. For instance, Alonso et al. (2008) would not accept the work from a user who has given the same answer for all the HITs he/she has done. Gelas et al. (2011) rejects the submission which has an empty transcription or has nonsense texts for an audio transcription task. Hsueh et al. (2009) identifies three dimensions (Noise, Ambiguity, and Confusion) and show that quality can be improved by eliminating the noisy annotations. On the other hand, Aroyo and Welty (2013a) proposes CrowdTruth to produce the gold standard leveraging the disagreement between workers. Furthermore, Nushi (2015) explores the impact of answer diversity and shows diversity can be utilised for quality prediction. Shaw et al. (2011) investigates different treatment conditions for a content analysis task and collected all the data, find that Punishment-agreement and Bayesian Truth Serum could help improve the worker performance. Generally speaking, once task requesters have collected needed judgments for their tasks, apart from using some straight-forward criteria (such as time spent, answer pattern, etc) to eliminate the potential spam data, they most likely seek to use effective algorithms on all the data to infer/predict correct answer for their tasks. Hence much research has been done in the past decade in this area which is reviewed in section 2.4.3.

### 2.1.4 Other Aspects of Microtasks Crowdsourcing

Although this thesis will only focus on classification task design and quality assessment method, other aspects that could potentially affect the quality of crowdsourcing work are worth considering. The crowd, as the actor in the crowdsourcing work, is fundamental to the quality of the final crowdsourced data. This section particularly reviews existing major work in this aspect about the crowd behaviour and their motivations.

Two of the crowdsourcing dimensions “Who” and “Why” are related to the crowd which makes it important to understand their motivations and design crowdsourcing in a way that can motivate the users. Researchers have explored different ways that can incentivize the crowd so that they can engage and contribute with high-quality work. Finding

---

the right motivation is obviously crucial to the success of a crowdsourcing system as the performance of the crowd worker directly affects the crowdsourcing result. Different motivations and rewards mechanisms have been studied in crowdsourcing tasks, including intrinsic and extrinsic motivations, gamification elements (points, badges, leaderboard, etc.), collaborations between crowd workers, or competitions among all participants.

- **Intrinsic and Extrinsic Motivation**: From the motivation point of view, existing studies have looked into both intrinsic and extrinsic aspects. People are intrinsically motivated when they feel it is something fun and enjoyable to do, and they are able to gain new knowledge, share knowledge or ideas, socialise with others, help others or contribute to the society (Shadbolt et al., 2013). Extrinsic motivations include monetary reward, recognitions, status, etc which could also be used to motivate participation. However, monetary reward has been shown to be not always the key element driving people to contribute voluntarily, instead, the inspiration of contributing something useful to the society seems more attractive (Howe, 2006). Cuel et al. (2011) has systematically investigated different motivation mechanisms for the crowd to create semantic content. The research takes a closer look at the intrinsic aspects which are directly related with how people would perform a task, as well as the extrinsic aspects which are unrelated to the task itself, and concludes that although crowdsourcing semantic creation might be time-consuming and error-prone, semi-automatic content creation seems to be the most promising approach which could reduce cost as well as encourage participation. Other research also shows different people or even the same people, their motivations could be different at different times, depending how they perceive the system at different times (Donath, 2014).

- **Beyond Monetary Rewards**: In the paid crowdsourcing platform, the incentive mechanisms are mainly driven by monetary rewards. However simply increasing payment does not necessarily lead to higher quality of result (Borst, 2010). As a result, researchers have looked at other motivators and investigated how they help influence the behavior and/or improve the performance of the crowd worker. Rogstadius et al. (2011) presents an empirical study using Amazon’s Mechanical Turk platform 4 to test how intrinsic motivation of “helping others” and “increased payment” affect the quality of the output from the crowd workers. Their result suggests that intrinsic motivation does improve the quality which is echoed by the study from Law et al. (2016). Law et al. (2016) investigate curiosity as an intrinsic motivational driver to retain workers. The study uses different design elements as curiosity stimuli to compare how users perform under different settings, and shows that with curiosity interventions, the number of completed tasks and quality of the answers from the crowd are both higher than the baseline where no curiosity stimuli was given.
Community, Collaboration and Contest: Howe (2006) examines the motivations why the crowd would engage in certain activities and points out as crowdsourcing heavily relies on collaboration of people from diverse background and dispersed locations, it is crucial to provide an online community where people feel attached to and are willing to participate. Most of the large online knowledge bases are contributed to by people volunteering in the corresponding community (Bryant et al., 2005; DeRose et al., 2008; Luczak-Rösch et al., 2014). Collaboration (Kittur and Kraut, 2010; Chamberlain, 2014; Rokicki et al., 2015) has been proved to be useful in the past. In “groupsourcing” (Chamberlain, 2014; Rokicki et al., 2015), tasks are completed by groups of users who are normally self-organised: Chamberlain (2014) uses social network groups in their experiment of image classification and show that the groupsourcing achieves comparable quality to the experts. Using a face recognition task, Rokicki et al. (2015) investigates how different team formation mechanisms and competition strategy can improve the cost efficiency. Feyisetan and Simperl (2016) explores how social pressure can motivate participants in a collaborative crowdsourcing context and show that social pressure is an effective approach to encourage more contributions. Contest (e.g. Netflix million dollar challenge or Climate CoLab) has also been used and proved to be effective. However, both intrinsic and extrinsic elements need to be considered when design a crowdsourcing contests: monetary reward, recognition, timely feedback, just to name a few of these elements (Zheng et al., 2011).

Love and Glory: One of the motivations mentioned by Malone et al. (2010) is known as “love and glory”. Love refers to the intrinsic satisfaction, fun and enjoyment people gained from participation in large group activities. Glory is another important incentive for most people, as people intrinsically desire to be recognised and respected by peers in their community. Many gamified elements such as points, levels, badges, leaderboards can be used as rewards mechanisms to represent “status” and give participants the sense of glory (Zichermann and Cunningham, 2011; Deterding et al., 2011). In the voluntary crowdsourcing contexts, aforementioned individual, collaborative or competitive motivations would be applicable, and probably more gamification mechanisms are the key to motivate users with “love and glory”. This can be evidenced by the ESP game (Von Ahn and Dabbish, 2004), as well as the huge amount of participants in the citizen science projects such as GalaxyZoo, Snapshot Serengeti, Cities at Night, FoldIt, EyeWire. Especially in the case of FoldIt and EyeWire, scientists have designed the system in a game-based format which makes them fun and enjoyable to play, at the same time the participants are motivated by the sense that they are helping science and their contribution is meaningful. These are considered as GWAP (‘games-with-a-purpose’) (Von Ahn et al., 2008) and have

21 http://netflixprize.com
22 https://climatecolab.org/
been shown useful as well in other contexts (Liem et al., 2011; OntoGame; Thaler et al., 2012; Simperl et al., 2013a). De Boer et al. (2012) present a mixed picture of balancing the gamification elements: They use a five level points mechanism along with badges in their experiments but find out in a context like digitisation of pluvial data from Africa that fun element are not the most important element, instead “benefit to Africa” seems to be the most motivational element based on the survey result from the participants.

- **Learning and Feedback**: In other volunteer projects, people are more engaged when they feel they can learn from the tasks they carry out or/and their input is valuable. Dontcheva et al. (2014) shows that the crowd’s engagement and performance could be improved by incorporating learning experience in the crowdsourcing tasks. Jones et al. (2013) also points out the learning experience provided to the participants in citizen science activities has a prominent effect on the participants’ attitude and behaviour. Such learning experience can be obtained by training and tutorials provided, as well as feedback given when they test their knowledge carrying out the corresponding tasks. Bigham et al. (2010) introduces the VizWiz system and point out many participants prefer to receive feedback on how they perform the tasks. Dow and Klemmer (2011) presents the Shepherd system where feedback can be managed and presented to the crowd workers, they conclude when feedback is provided, the user performance is higher than when no feedback is available. Abad et al. (2013) investigates real-time feedback in speech transcription, comparing two different type of task-specific feedback: a) expert feedback or turker generated feedback with b) no feedback baseline case using experiments on Amazon Mechanical Turk. Abad et al. (2013) shows that providing feedback helps improve the quality of the task and crowd workers achieve higher accuracy, performing best when expert feedback is given. This research echo the theory in (Kaufmann et al., 2011) where feedback can be indirect feedback (part of the social motivation) or direct feedback (part of the enjoyment based motivations), and motivate users respectively by presenting an overview of the performance of other users or providing a sense of achievement during/after the task execution. Bernaschina et al. (2015) looks at feedback from a different angle by giving the crowd worker feedback on how much they have earned, tasks remaining to be completed, and additional bonus to be granted based on the correct answers they have given. Bernaschina et al. (2015) finds out how average accuracy is improved when users are paid in batch, and achieve higher accuracy when feedback is present but seems to have a lower accuracy when the payment is by a piece of work even with feedback. What type of feedback to provide, how to present it, how to integrate it in a dynamic and interactive crowdsourcing environment are the areas still not sufficiently investigated (Roy et al., 2013).
2.2 Classification

In this section, it starts by introducing the general idea of classification and its main dimensions. The role of human beings in the classification process, and how crowdsourcing can be used for classification tasks is then reviewed. Following that, two common classification areas, entity classification and image classification, will be used as representative examples to help us understand the specifics about classification for these two areas. How classification has been done in different ways via manual, automatic or hybrid workflows by incorporating crowd into the classification process is reviewed. At last, it summarises the challenges in these areas that this research is about to tackle.

2.2.1 Classification Basics

Classification is about putting objects into corresponding categories based on their similarities. Categories are normally designated by names and are organized in a taxonomy system where categories are related to each other by inclusion or exclusion (Rosch et al., 1976). For instance, Dog, Cat, Animal are all categories to represent real world objects that have similar or different attributes. Rosch and Lloyd (1978) points out that to categorise something it is not only to find the similarity to other objects in the same category, but also to differentiate from other objects that are not in that category. This corresponds to the dimension of the category system, which contains both a vertical (“level of the inclusiveness”) and a horizontal (“segmentation of categories at the same level of inclusiveness”) dimension (Rosch et al., 1976; Rosch and Lloyd, 1978). Categories in vertical dimension vary in the level of details they provide within that abstract level of category, for instance, when we categorise a thing as Animal, or Dog, or German Shepherd Dog, they are all correct but different level of detail information is provided which then can be mirrored to the structure of attributes perceived in the real world. On the other hand, the horizontal dimension of categories emphasizes the distinctiveness among categories.

Human beings have the intrinsic ability to classify objects and are good at using structure to understand the objects of the world although each person may have a different perception based on their personal experience (Batley, 2014). To simplify the world and make sense of the objects in the real world, classification is used almost everywhere using different classification schemes ranging from general classification in differentiating natural objects, to specific classification based on personal experience or special contexts, to domain-specific classifications. For instance, three persons all have recognised a “dog”, one may classify it as a “dangerous animal” while the other categorizes it as a “friendly animal” (Batley, 2014), and another one can specifically identify it as a “German Shepherd Dog”. Rosch and Lloyd (1978) also point out that “categories tend to
become defined in terms of prototypes or prototypical instances that contain the attributes most representative of items inside and least representative of items outside the category”. When a human being is classifying an object, cognitively they may compare the object to a known representative object to see how similar they are, or match the object’s attributes to the description of specific categories. It is clear from the previous example that when asking human beings to classify a thing, several aspects need to be explicitly specified to achieve consistent classification: 1) a classification scheme (the pre-defined categories or category structure) with descriptions of defined categories 2) level of details expected, is it the most abstract level is enough? or the most specific category is expected? 3) How many details or contexts are provided about the thing being classified? For example, if details of a dog describing it as “soft, mild tempered, playful” are given, it helps to classify it more accurately even if we may not have corresponding knowledge in advance. 4) a representative example might be helpful in giving an intuitive impression of the attributes defining that category and thus help to associate the thing to be classified to the specific category.

Even though classification has been an activity since ancient times\textsuperscript{23}, there are lots of new or unknown objects that need to be classified for human beings to better understand the world and develop knowledge around them. Nowadays, there are more and more needs in classification both for scientific research\textsuperscript{24} and improving daily lives of human beings\textsuperscript{25}. With crowdsourcing becoming popular and promising in the recent decade, it is no surprise that classification tasks are one of the most popular types of crowdsourcing tasks. In the crowdsourcing context, using the crowd to help classify objects means to ask the crowd to classify data into a set of pre-defined related categories. The pre-defined categories can be a binary class or multiple classes. For instance, “Is this an apple?” can be treated as a binary classification case where the categories are \textit{yes}, \textit{no} to indicate the category of the data being evaluated is “an apple” or “not an apple”. “Identify the shape of the object” can be a binary classification, or a multiple-class classification depending on the categories were given. For example, if the categories are \textit{rectangle}, \textit{not rectangle} it is a binary classification; It turns to a multi-class classification if the categories contain more than two categories, such as \textit{rectangle}, \textit{triangle}, \textit{eclipse}, ..., \textit{line}. Difficulty and complexity of classification tasks can vary (Alfonso-Reese et al., 2002a). In general, the data to be classified can have different forms: texts, images, or audios, etc. The data, along with the descriptive question and the corresponding pre-defined categories are then presented to the crowd. Normally classifications from multiple crowd workers are collected and aggregated afterward to decide the correct category. This may sound straightforward but is challenging when either completely relying on machine or human to do the classification. Certain types of classifications have specific characteristics which make them difficult to be done simply in one approach, such as entity classification and

\textsuperscript{23}https://www.sciencelearn.org.nz/resources/1438-classification-system
\textsuperscript{24}https://blog.zooniverse.org/2017/06/
\textsuperscript{25}http://www.scienceclarified.com/everyday/Real-Life-Biology-Vol-2/
Taxonomy-Real-life-applications.html
image classification which will be elaborated in section 2.2.2 and section 2.2.3. The following paragraphs elaborate the classification methods currently used in these two fields and the challenges in each of these areas.

### 2.2.2 Entity Classification

Entity classification refers to the process of identifying the class for an entity. For instance, “Berlin” is a Place/Location in general and a PopulatedPlace based on DBpedia’s ontology. Generally, entity classification can be carried out in different ways, ranging from manual annotation by experts from the corresponding domain, over hybrid approaches where human input and machine algorithms are combined, to the tools for purely automatic classification. As expert classification is a costly and time-consuming task, automatic entity classification traditionally has been part of the NER (Named Entity Recognition and Classification) research where the entity is first automatically identified and then classified according to a number of categories (Niu et al., 2003; Wu et al., 2006; Collins and Singer, 1999; Kim et al., 2002b). Unfortunately, such automatic algorithms require predefined concept-based seeds for training and manually defined rules, which can become very complicated if a large number of classes were involved. As a consequence, the classification ability is typically limited to a relatively small number of classes such as “Person”, “Location”, “Time” and “Organisation”. Those approaches are often combined with machine learning algorithms (e.g., Support Vector Machine, Naive Bayesian) or ensembles trained on large pre-annotated datasets. However, all discussed techniques are working with a certain error margin and hence their output still requires to be manually re-examined and further classified where crowdsourcing could help. There are a number of existing tools and APIs such as DBpedia Spotlight, Dandelion, Alchemy API, Open Calais, and NERD that can be used to classify entities based on a predefined ontology, but they very often fail to produce a type for some of the entities.

Entity classification by combining human and machine seems to be a promising alternative (Costa et al., 2011; Simpson et al., 2013; Snow et al., 2008; Von Ahn, 2006; Wang et al., 2012). Some studies show that it is possible to combine automatic prediction methods (Bayesian/Generative probabilistic models) with additional input from the crowd to improve output accuracy (Simpson et al., 2013; Loni et al., 2014; Hare et al., 2013; Sheng et al., 2008). Wang et al. (2012) have also shown the performance advantage of the hybrid approach when compared with fully automatic methods. There

---

26 [http://dbpedia.org/page/Berlin](http://dbpedia.org/page/Berlin)
27 [http://dbpedia.org/ontology/Location](http://dbpedia.org/ontology/Location)
28 [http://dbpedia.org/ontology/PopulatedPlace](http://dbpedia.org/ontology/PopulatedPlace)
30 [http://dandelion.eu/semantic-text/entity-extraction-demo/](http://dandelion.eu/semantic-text/entity-extraction-demo/)
exist basically two types of hybrid approaches: The first is based on collecting a lot of annotations from the crowd and using the collected labels to train machine learning algorithms in order to achieve better classification quality. An example of such an approach is the ESP game by Von Ahn (2006) for gamification-based images labelling. The other approach is to use a machine to narrow down the possible options and then employ the crowd to validate or choose the best matching one. As an example, the work presented by Demartini et al. (2013) employs a machine-based algorithm to classify entities along with calculating a confidence score. The authors suggest that the label crowdsourcing is required only for entities with low confidence scores produced by the classifier. Nevertheless, existing research tends to focus on classifying the entities into “basic level” (Rosch et al. (1976) defined “the basic level as the level that has the highest degree of cue validity”). There is a lack of studies in classifying entities to their most specific categories which raises challenges on how this can be effectively done via hybrid approach, the factors to consider during task and workflow design to allow the crowd to be able to locate the proper category through a classification hierarchy.

2.2.3 Image Classification

Image classification is one of the important tasks in computer vision and has been mainly tackled using automatic approaches which normally include image capturing, pre-processing, object detection, object segmentation, feature abstraction and image classification (Kamavisdar et al., 2013). From the perspective of using training samples, image classification can be done by unsupervised classification and supervised classification (Lu and Weng, 2007; Kamavisdar et al., 2013). In the unsupervised classification case, a large number of unknown pixels from the image is checked and divided into different natural groups, which then need to be compared to reference data to determine the identity and informational values of the spectral classes (Denniss, 1995). For the supervised classification, samples with known classes are used as training sets. A statistical characterisation of the reflectance (aka, “signature analysis”) for each information class is developed and the image is then classified by examining the reflectance for each pixel and deciding on which of the signatures it resembles most (Eastman, 1996). We have seen lots of advanced approaches applied in the image classification area (Zhang, 2000; Omo-irabor, 2016), such as neural network classifiers, maximum likelihood, support vector machine, iterative self-organising data analysis technique, K-means, decision tree, etc. In these supervised approaches, labels are needed. However, labels are not always available and are expensive to get if using the help from experts. In some cases, such as for web images, it is possible to automatically collect/extract labels from the surrounding texts (Gong et al., 2014) but it is not the case for most of the image classification tasks as there are no relevant descriptive texts associated with them.
In recent years, researchers have started to utilise the help of the crowd in the classification process. The ESP game (Von Ahn and Dabbish, 2004) is one example of image labelling by the crowd in a gamified context, which not only helps classify the images but also enriches them to allow image searches, and possibly applications using the collected labels to provide useful textual descriptions for the images. In a similar image search context, Yan et al. (2010) uses the real-time crowd to validate the automatic image search results before the search result is returned to the user. CrowdSearch (Yan et al., 2010) focuses on the top-5 candidate images returned from the search process, which has been optimised to extract features from images and produce a ranked list. Such a hybrid approach to combine both crowd input and machine algorithms seems to be promising in producing a better result and is widely adopted by later research. Loni et al. (2014) investigates two crowdsourcing tasks that help validate(classify) whether a given image is related to the clothing and fashion domain, and its type. They show how the classification can be improved to use input from the crowd in an optimal way by exploiting information from other sources at the same time. Ramchurn et al. (2015) presents an approach where the crowd can classify images from the disaster area to help identify the area where immediate response is needed, with the use of machine learning algorithms to process the collected data from the crowd and predict the locations of emergencies. However, there are classification tasks that involve more classification steps than the single step classification, and it is unclear how different task and workflow design might affect the classification result in such cases. At the same time, it is quite common that collected data from the crowd is of various quality (Kittur et al., 2008). This is also why lots of research focus on developing inference algorithms to effectively aggregate the crowdsourced classification, as described in section 2.4.3. However, there is a little research in how answer inference could be applied in complex classification scenarios where answers for multiple interdependent classification questions (instead of one single classification question) are needed to be aggregated. These correspond to the first two research questions of this study, and will be explored in Chapter 4 and 5.

The classification task is also one of the most important types of tasks in citizen science projects such as Zooniverse and Stars4all. Citizen science project Backyard Worlds co-founder, Aaron Meisner, once said, “Professional researchers routinely use supercomputers and machine learning, but there’s still no substitute for the human eye when it comes to recognising subtle motions in astronomical images,” which is true in many image related classification tasks. Most of the citizen science projects involve huge amount of participants such as GalaxyZoo, Snapshot Serengeti, Cities at Night, FoldIt, EyeWire.

---

34https://blog.zooniverse.org/2013/06/20/how-the-zooniverse-works-the-domain-model/
35https://newsbeezer.com/ireland/citizen-scientists-are-invited-to-join-a-quest-for-new-worlds/
36http://citiesatnight.org/
Each of these projects contains a number of diverse projects from different scientific research domains such as space, nature, climate, medicine, humanity and light pollution. Most of the classification is based on images taken from space, wildlife park or nature, some are audios captured under the sea or in the forest. These projects were done by asking the hundreds of thousands of volunteers to help classify the objects. The classifications from multiple volunteers are then aggregated by scientists using different approaches, ranging from relatively simple majority voting based mechanisms to sophisticated machine learning and statistical inference algorithms (see section 2.4.3). This research will take a close look at these existing projects and identify the important factors that affect the classification quality.

2.3 Classification Task via Microtask Crowdsourcing

In recent years, researchers have successfully applied human computation and crowdsourcing to a variety of scenarios and largely used “classification” tasks. The idea is to decompose tasks with higher complexity into smaller sub-tasks (Shahaf and Horvitz, 2010) such that each of those tasks can be solved by non-expert crowd workers. Due to the decentralised and diverse nature of possible participants in crowdsourcing work, data validation and quality control have been an essential topic explored by many previous researchers, resulting in challenges in workflow and task design.

Both task and workflow are not terms specific to Crowdsourcing. Task refers to a piece of work to be executed and workflow refers to a series of steps to complete a task\textsuperscript{37}. Depending on the complexity of the task, a task might be decomposed into smaller fine-granular microtask to be carried out by multiple users in parallel or sequentially. When we talk about the task in general, we mean the ultimate goal and usually use tasks, sub-tasks, and microtasks interchangeably referring to the decomposed single unit of work. A task is represented in question format, and the annotations or answers are the data collected through crowdsourcing process. The simplest form of a workflow can include one question in one step, with one single type of microtask. For instance, identifying a person’s name from a tweet, or tagging an image with an English word. To tackle the more complex problem, the task is normally split into smaller tasks (microtask) which are easy to be carried out by non-experts. These microtasks can be either independent or dependent. For instance, the find-fix-verify workflow proposed by Bernstein et al. (2010) is used to solve the complex problem of proofreading by dividing it into three relevant steps. Studies on task and workflow design are always interlinked for complex tasks in crowdsourcing, as the way tasks are decomposed and chained are inseparable.

\textsuperscript{37}http://searchcio.techtarget.com/definition/workflow
2.3.1 Task Design

Existing studies have looked at tasks and workflows in various areas and investigated the corresponding task design, such as using the crowd to classify or annotate video, classify or search images, analyse texts (e.g. topic, sentiment), evaluate truthfulness of news/article/statement, or to help in disaster situations which might require a real-time response. As my research centered on the classification microtask crowdsourcing, the following related work is reviewed mainly from the classification task design point of view.

Early research in crowdsourcing from Kittur et al. (2008) investigates the major challenges faced by microtask crowdsourcing. In particular, the study presents recommendations on how task formulation can help identify low-quality input from the crowd by including a quantitative verifiable question, which has been accepted in later studies (Sarasua et al., 2012; Kucherbaev et al., 2014). Heer and Bostock (2010) uses the crowd to evaluate visualisation design, though the task is replicated from previous research, also concludes the same as Kittur et al. (2008) that the combination of verifiable question and test questions can achieve a quality result. Using test questions to filter the crowd workers before allowing the user to do any tasks, has been proved to be effective in improving quality by other researchers (Thaler et al., 2012; Zhuang et al., 2015; Zheng et al., 2017). Research in leveraging the crowd in semantic web, ontology engineering and linked data management (Eckert et al., 2010b; Simperl et al., 2011a) area presents feasibility of transforming some of the semantic data management processes into carefully designed classification tasks such as verifying the “sameAs” case of two identifiers/resources in Linked Open Data, classification of resources, and evaluating concept relatedness (by 5 scales) and generality (choosing from 4 options) for constructing concept hierarchy purpose in the InPhO (Indiana Philosophy Ontology) project38. These studies are mainly one simple question microtasks which are then integrated with other automatic steps of the semantic data management framework or system. At the same time, they exhibit how the human input part can nicely fit into the system and presents hybrid workflows. Named entity recognition (Finin et al., 2010; Feyisetan et al., 2015b) and sentiment analysis (Hsueh et al., 2009; Liu et al., 2012) are popular tasks that have been piloted in microtask crowdsourcing and are normally limited to a small number of fixed options/categories, with challenges unsolved in the presence of a large number of categories. Feyisetan et al. (2015b) looks into the factors that may affect the quality of named entity recognition task, including the number of entities, the type of entities and the length of the texts (micropost, twitter). Feyisetan et al. (2015b) also observed an interesting user behavior that the users tend to annotate the center of the texts rather than other parts.

38http://inpho.cogs.indiana.edu
Sigurdsson et al. (2016) conducts a study with video chips annotation and conclude the optimal strategy to get high-quality annotations would be using many imperfect simple questions, such as many binary classification questions instead of complex questions which have many available options. Their research also compared the effects of having few-questions, many-questions, grouping similar visual concepts on the interface. Salisbury et al. (2015) investigates a similar video annotation task but in a real-time crowdsourcing context which uses the pre-hired crowd as recommended by Bernstein et al. (2011). Bigham et al. (2010) introduces “quikTurkit” to pre-hire crowd workers so that visual questions can be answered in a near real-time fashion and the VizWiz system presented in their work not only shows different ways a visual question can be asked but also an interesting hybrid process where initial data is from blind people who took a picture and recorded an audio question, which are automatically converted to HITs (Human Intelligent Tasks), answered by pre-recruited crowd workers via quick-Turkit. Yan et al. (2010) uses simple Yes/No questions in Amazon Mechanical Turk to validate whether the image search result matches the query, and at the same time consider time constraints to obtain near real-time answer by allowing setting deadline in the task. Real-time crowdsourcing is particularly useful in disaster management (Zook et al., 2010; Ramchurn et al., 2015). Ramchurn et al. (2015) asks the crowd to classify images or texts into several emergency types in disaster response, combine the crowd’s input using machine learning algorithms to predict the emergency area and then present it to the crowd for further labelling. Crowdsourcing-based verily offers the ability to validate facts as well as verifying information during disasters but does not restrict the question to be posted in a close-ended way. However, the task it needs the crowd to verify is designed in a way structured information could be extracted and the main input is a binary question asking the crowd to confirm whether the image is real or not, with mandatory field for explanation texts to state why and optional fields for image, video, link or descriptive texts as evidence. More recent work look at the design of the task instructions and related training process to the crowd (Liu et al., 2016), and shows that gated instructions can help improve the quality of annotation and labelling.

2.3.2 Workflow Design

Workflow design is an integral part of complex task crowdsourcing as the tasks are decomposed into smaller microtasks to make them suitable for non-expertise crowd workers. The tasks need to be carried out in parallel or a series of sequentially dependent or independent steps. Most previous studies around crowdsourcing workflows have focused on the design of the workflows and have shown that a particular type of workflow can be crowdsourced effectively (in terms of the accuracy of outputs, budget, time, etc) (Little et al., 2009; Bernstein et al., 2010; Tran-Thanh et al., 2015). The find-fix-verify pattern proposed by Bernstein et al. (2010) has been widely accepted and proved to be a best
practice (Sarasua et al., 2012; Lin et al., 2012; Acosta et al., 2013a; Ramchurn et al., 2013; Karger et al., 2014) in terms of task and workflow design. The find-fix-verify (FFV) was firstly brought up when Bernstein et al. (2010) designs tasks to improve writing by involving multiple crowd workers which present a phased approach to first find the error to correct or the sentence to be shortened, then a set of crowd workers try to fix the problems identified from the first stage, which is then verified in the third phase. Later studies built upon the FFV workflow pattern and investigate more complex workflows in different contexts. Sarasua et al. (2012) investigates microtask crowdsourcing in ontology alignment and automatically publish crowdsourcing tasks based on the automatically found mapping pair candidates between two different ontologies. Hybrid workflows using microtasks as a component of the ontology engineering process seem to be quite effective based on previous studies (Noy et al., 2013a; Acosta et al., 2013a). Yan et al. (2010) combines the automatic image search results with an adaptive probability-based algorithm to decide which result needs to be validated and automatically generate the tasks to be carried by the crowd. Ramchurn et al. (2013) presents a workflow where FFV pattern is applied and corresponding new tasks are automatically generated based on current phase and related metrics, complex drawing evacuation routes task is decomposed into smaller microtasks which are completed by multiple crowd workers, to identify a building, verify a building, identify a route, verify a route and verify the completion respectively.

There are a few studies on workflows which are not inspired by the FFV pattern. Little et al. (2009) presents the iterative approach allowing easily planning the parallel tasks and evaluation among peers. Hu et al. (2010) uses iterative collaboration in the translation process that also combines with machine-translated results that are reviewed and enhanced by the crowd. Kittur et al. (2011) proposes the CrowdForge framework to facilitate complex and interdependent tasks using microtasks. The framework provides a toolkit allowing task requesters to combine the task primitives (partitions, map, and reduce) to manage the complex tasks crowdsourcing in stages: 1) using a pre-specified partition to decompose tasks to smaller subtasks, 2) using a flow to control sequences of the decomposed subtasks, as well as adding quality control, 3) automatically aggregating result. Sigurdsson et al. (2016) proposes that simultaneously asking multiple binary questions is more effective based on the empirical study with video clips annotation. Some research in the workflow design area involves algorithmically evaluating results based on certain criteria to dynamically allocate tasks for better results or lower costs (Demartini et al., 2012; Karger et al., 2014; Tran-Thanh et al., 2015). Dynamic task allocation is not the focus of this work. However, the algorithms used to evaluate the result, will be reviewed in section 2.4.3. The workflow design is a rather complicated topic itself which is normally either cost, time or quality focused. From classification perspective, a workflow needs to consider the task division, the involvement of auto(semi-auto) steps, type of microtasks, the order of microtasks, the dependency of microtasks, criteria on the transition between microtasks, and task assignment.
<table>
<thead>
<tr>
<th>Focus</th>
<th>Main Objective</th>
<th>Related Work</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>Mitigate malicious input</td>
<td>Sarasua et al. (2012) &lt;br&gt; Kucherbaev et al. (2014) &lt;br&gt; Heer and Bostock (2010)</td>
<td>using verifiable questions during the task execution</td>
</tr>
<tr>
<td>Task</td>
<td>Filter workers with good quality input</td>
<td>Kittur et al. (2008) &lt;br&gt; Thaler et al. (2012) &lt;br&gt; Zhuang et al. (2015) &lt;br&gt; Zheng et al. (2017)</td>
<td>users have to pass test questions before they can continue with the task</td>
</tr>
<tr>
<td>Task</td>
<td>Feasibility in using microtasks for domain-specific areas which usually require experts to do, such as semantic web or named entity recognition</td>
<td>Eckert et al. (2010b) &lt;br&gt; Finin et al. (2010) &lt;br&gt; Simperl et al. (2011a) &lt;br&gt; Feyisetan et al. (2015b) &lt;br&gt; Hsueh et al. (2009) &lt;br&gt; Liu et al. (2012)</td>
<td>using simple one-step microtask which is then integrated with other automatic steps (e.g. of the semantic data management framework or system) can be promising.</td>
</tr>
<tr>
<td>Task</td>
<td>Design choice: one question with multiple-options, or multiple binary questions</td>
<td>Sigurdsson et al. (2016)</td>
<td>using many imperfect simple questions to get high-quality annotations</td>
</tr>
<tr>
<td>Task</td>
<td>Obtain real-time answers for microtasks</td>
<td>Bigham et al. (2010) &lt;br&gt; Yan et al. (2010) &lt;br&gt; Salisbury et al. (2015) &lt;br&gt; Ramchurn et al. (2015)</td>
<td>pre-hire crowd workers, make task as simple as possible, and set deadline for task</td>
</tr>
<tr>
<td>Workflow</td>
<td>Feasibility of having the crowd to do complex task which are usually done by experts, such as writing</td>
<td>Bernstein et al. (2010) &lt;br&gt; Sarasua et al. (2012) &lt;br&gt; Lin et al. (2012) &lt;br&gt; Acosta et al. (2013a) &lt;br&gt; Ramchurn et al. (2013) &lt;br&gt; Karger et al. (2014)</td>
<td>find-fix-verify</td>
</tr>
<tr>
<td>Workflow</td>
<td>Hybrid workflow to tackle specific domain such as tasks in ontology alignment/engineering</td>
<td>Sarasua et al. (2012) &lt;br&gt; Noy et al. (2013a) &lt;br&gt; Acosta et al. (2013a)</td>
<td>using automatic method to generate the initial tasks which are then automatically promoted to crowd users for further processing (e.g. validating the mapped candidates from two ontologies, or confirming the entity’s class)</td>
</tr>
</tbody>
</table>
Chapter 2 Background

<table>
<thead>
<tr>
<th>Workflow</th>
<th>Feasibility of using the crowd to tackle “Create” tasks (Malone et al., 2010) such as translation and text improvement</th>
<th>Little et al. (2009) Hu et al. (2010)</th>
<th>split the task into small parts, post them as independent tasks, and pass on the finished tasks to different crowd workers to iteratively work on them.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow</td>
<td>Achieve higher quality or lower cost</td>
<td>Demartini et al. (2012) Karger et al. (2014) Tran-Thanh et al. (2015)</td>
<td>algorithmically evaluating results based on certain criteria to dynamically allocate task</td>
</tr>
</tbody>
</table>

Table 2.1: Representational task/workflow design

2.4 Classification Quality Assessment

This section reviews quality dimensions (section 2.4.1) and quality metrics (section 2.4.2) used in the crowdsourcing classification context, and takes a look at the methods that can be utilised to assess the quality of crowdsourcing classification result, covering existing algorithms that are used to infer correct answers in section 2.4.3.

2.4.1 Quality Dimensions

Quality has many dimensions and can be measured in different ways (Bergdahl et al., 2007; Wang et al., 1995, 2013b; Tu and Wang, 1993; Wang et al., 1993; Pipino et al., 2002; Lee et al., 2002; Batini et al., 2009; Ipeirotis et al., 2010; Zaveri et al., 2013). Bal lou and Pazer (1985) identified “accuracy, timeliness, completeness, consistency” as the important data quality dimensions. Wang et al. (1993) presents a data quality modelling methodology to facilitate quality control and systems design, in which a sharing standard and generalisable definition of data quality parameters, indicators, dimensions (including the commonly recognized important dimensions “completeness, timeliness, accuracy, and interpretability”) are required (Wang et al., 1995). Kahn et al. (2002) focuses on information quality and defined 16 quality dimensions (Figure 2.1. Lee et al. (2002) categorises the information quality dimensions from an academic’s point of view (Figure 2.2) into four main classes: Intrinsic Information Quality, Contextual Information Quality, Representational Information Quality and Accessibility Information Quality. Batini et al. (2009) surveys existing literature and summarised a list of different dimensions concerning data quality: Accuracy, Completeness, Consistency, Time-related dimensions including Currency, Volatility, and Timeliness. Bizer and Cyganiak (2009) focuses on six core linked data quality dimensions that are similar to those in Batini
et al. (2009), including Accuracy, Completeness, Consistency, Timeless, Uniqueness and Validity. Zaveri et al. (2013) has a comprehensive review on Linked data quality dimensions and assessment methods. They identified 18 quality dimensions which can be categorised into four general groups: Accessibility dimensions (availability, licensing, interlinking, security and performance), Intrinsic dimensions (syntactic validity, semantic accuracy, consistency, conciseness, completeness), Contextual dimensions (relevancy, trustworthiness, understandability, timeliness) and Representational dimensions (representational conciseness, interoperability, interpretability, versatility). It is worth noting that “accuracy” (“correctness”) is agreed by all the previous studies as an important dimension, which is intrinsic to the data quality. Most of the other aforementioned dimensions are applicable to the dataset produced from crowdsourcing activities, such as accessibility, interpretability, consistency and completeness. These aspects of the crowdsourcing dataset quality concerns more on how task requester make their data in a good state to be shareable and reusable, which is an emerging research area to use semantic web technologies to describe crowdsourcing activities and capture crowdsourced data (Seifert et al., 2013; Silva and Ramos, 2014; Sarasua et al., 2015; Sivula and Kantola, 2015; Alabduljabbar and Al-dossari, 2016). In this study on quality assessment of the crowdsourcing result, the focus is on how task and workflow design in crowdsourcing could affect the intrinsic quality (Lee et al. (2002)) of the collected crowdsourcing data.

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Definitions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accessibility</td>
<td>the extent to which information is available, or easily and quickly retrievable</td>
</tr>
<tr>
<td>Appropriate Amount of</td>
<td>the extent to which the volume of information is appropriate for the task at hand</td>
</tr>
<tr>
<td>Information</td>
<td></td>
</tr>
<tr>
<td>Relievability</td>
<td>the extent to which information is regarded as true and credible</td>
</tr>
<tr>
<td>Completeness</td>
<td>the extent to which information is not missing and is of sufficient breadth and depth for the task at hand</td>
</tr>
<tr>
<td>Concise Representation</td>
<td>the extent to which information is compactly represented</td>
</tr>
<tr>
<td>Consistent Representation</td>
<td>the extent to which information is presented in the same format</td>
</tr>
<tr>
<td>format</td>
<td></td>
</tr>
<tr>
<td>Ease of Manipulation</td>
<td>the extent to which information is easy to manipulate and apply to different tasks</td>
</tr>
<tr>
<td>Free-of-Error</td>
<td>the extent to which information is correct and reliable</td>
</tr>
<tr>
<td>Interpretability</td>
<td>the extent to which information is in appropriate languages, symbols, and units, and the definitions are clear</td>
</tr>
<tr>
<td>Objectivity</td>
<td>the extent to which information is unbiased, unprejudiced, and impartial</td>
</tr>
<tr>
<td>Relevancy</td>
<td>the extent to which information is applicable and helpful for the task at hand</td>
</tr>
<tr>
<td>Reputation</td>
<td>the extent to which information is highly regarded in terms of its source or content</td>
</tr>
<tr>
<td>Security</td>
<td>the extent to which access to information is restricted appropriately to maintain its security</td>
</tr>
<tr>
<td>Timeliness</td>
<td>the extent to which the information is sufficiently up-to-date for the task at hand</td>
</tr>
<tr>
<td>Understandability</td>
<td>the extent to which information is easily comprehended</td>
</tr>
<tr>
<td>Value-Added</td>
<td>the extent to which information is beneficial and provides advantages from its use</td>
</tr>
</tbody>
</table>

Figure 2.1: Dimensions of information quality. Kahn et al. (2002)

In the microtask crowdsourcing context, achieving a good quality result is one of the major goals. Therefore, when we talk about quality, it generally means the quality of the data collected from the crowd, which is mainly focusing on the intrinsic data dimensions.
Chapter 2 Background

Figure 2.2: Mapping the IQ dimensions into the PSP/IQ model. Lee et al. (2002)

To be specific, for the classification microtasks, the quality of the result refers to how good the overall classifications are, which is a data-value centric dimension to reflect how accurate the classifications are. In this work, if not specially specified, when referring to quality of the input/answer/data/result produced from the microtask crowdsourcing activities, it means Accuracy – “The degree to which data values correctly represent the real world facts” (Zaveri et al., 2013); similar definition in Bizer and Cyganiak (2009) as “The degree to which data correctly describes the “real world” object or event being described.”; definition in science (JCGM, 2008) as “closeness of agreement between a measured quantity value and a true quantity value of a measurand”. In a similar vein, when we talk about the quality of a crowd worker’s work which is a reflection of the worker’s performance, we refer to how good the crowd worker is in answering the classification question correctly.

2.4.2 Quality Metrics

Studies in performance evaluation in science normally use various measures apart from the widely used and most intuitive accuracy metric. Formulas for some of these measures need to consider true positive(TP), true negative(TN), false positive(FP), false negative(FN), true positive rate(TPR) and false positive rate(FPR) separately. A brief definition of these commonly used metrics is listed below in the classification context.

In terms of a classification task, existing work in quality assessment of the aggregation algorithms mostly use the accuracy metric (Khattach and Salleb-Aouissi, 2011; Kamar et al., 2012a; Quoc Viet Hung et al., 2013; Sheshadri and Lease, 2013; Zhang et al., 2013).
<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>It is a ratio of the correctly classified objects to the total number of objects being classified. It can be calculated by (TP+TN)/(TP+TN+FP+FN).</td>
</tr>
<tr>
<td>Precision</td>
<td>It is a measurement to describe the random error, and it reflects how reproducible/precise the classification is. Precision can be calculated by TP/(TP+FP).</td>
</tr>
<tr>
<td>Recall</td>
<td>Also known as “Sensitivity”. It is the ratio of correctly classified positive objects to all the objects in that class. TP/(TP+FN), this is also known as the TPR.</td>
</tr>
<tr>
<td>FPR</td>
<td>It is the ratio of false positives to the total number of negatives. FP/(FP+TN), this is also known as “expectancy”.</td>
</tr>
<tr>
<td>F1 score</td>
<td>It is the weighted average of both precision and recall. It is calculated as 2*(Precision*Recall)/(Precision+Recall)</td>
</tr>
<tr>
<td>ROC</td>
<td>Receiver Operating Characteristic curve is a graph plotted based on the TPR and FPR for identifying different cut-off points of a specific parameter. The area under the ROC curve, AUC, is normally used to measure how well a parameter can distinguish between two different groups.</td>
</tr>
<tr>
<td>RMSE/RMSD</td>
<td>Root Mean Square Error/Deviation is frequently used to measure the difference between values classified by a model, classifier, aggregated result from crowdsourcing data and the actual class.</td>
</tr>
</tbody>
</table>

2017; Zheng et al., 2017). Some research also uses precision/recall (Hung et al., 2015; Salisbury et al., 2015; Yin et al., 2016; Zhang et al., 2017) or F1 score (Zheng et al., 2017), some work uses accuracy, false positive and false negative (Wang et al., 2013a), while other work use ROC (Zheng et al., 2017) or RMSE (Bachrach et al., 2012b).

There are a few studies looking at the crowd workers' input from a different angle by using disagreement or diversity metric which is not directly evaluating the input quality. However, it is useful in providing a better understanding of how the quality could have been improved, or revealing interesting properties of the objects under investigation. Finin et al. (2010) measures worker agreement in named entity recognition tasks in which the crowd is asked to choose one of four categories (Person, Place, Organisation, None), which has shown the proposed agreement metric functions well as a quality metric to measure the classification correctness. Their study also uses the disagreement on the tagging of a particular object (word in this case) to decide whether this word is a good example to be used as training data. Aroyo and Welty (2013a) also uses disagreement as a measurement metric to understand sentence clarity and identify spammers. Gurari and Grauman (2016b) investigates different approaches that can be used to predict whether users tend to agree or disagree on the answers to visual questions and present the CrowdVerge system to automatically output predictive cues indicating whether answer from the crowd will converge or diverge.

In this study, by default, when talking about the quality of classification result (sometimes called “answers to the questions” presented in microtasks, or “data collected from the crowd”), the goal is to measure how good the answers are as a whole compared with
Chapter 2 Background

the gold standard, inferred or predicted correct answers and present by percentage of correctly classified objects. Gold standards are only used in our study for evaluating quality metrics, it is not meant to be part of the classification workflow, hence the time and cost taken to obtain/create the gold standard is not specially considered. In the context of this research, the performance of a worker, a specific workflow or an algorithm can be measured using a similar quality metric. For an individual worker, his performance can be measured by the percent of correctly classified objects against all the objects he has classified. For a workflow, its performance is implicitly reflected by the quality of classification achieved. For an algorithm, its performance will be how well it predicted the correct answer when compared with the given gold standards. During the research of specific cases, additional measures might be introduced and will be explained in the corresponding Chapter 4 and 5.

2.4.3 Inference Algorithms

In the crowdsourcing context, the ground truth is not usually available. In order to assess the quality of the result, we first need to understand what algorithms or mechanisms can be used to infer or predict the correct answer based on all the input from the crowd workers. Correspondingly each existing different algorithm has been studied by researchers and evaluated its performance in various contexts. This section mainly takes a look at these existing algorithms and provides an overview of the available benchmarks that can be used for assessing algorithm performance.

Researchers have proposed so-called inference algorithms, mathematical models that can automatically infer the correct solution to a given problem from a solution space defined by the crowd. For example, Ipeirotis et al. (2010) presents an algorithm that assesses the performance of crowd workers and exploits this information to estimate the quality of answers on Mechanical Turk. Karger et al. (2011) proposes to use message passing to determine correct answers. Bachrach et al. (2012b) usez a Bayesian graphical model to grade test answers in scenarios where ground truth cannot be made available. Whitehill et al. (2009) follows an expectation maximisation approach to identify correct classifications, depending on the expertise of the workers and the level of difficulty of the task. In the citizen science project Galaxy Zoo Supernovae, crowd answers were analysed using a Bayesian generalisation of the same expectation maximisation idea (Simpson et al., 2011). Difallah et al. (2015) compiles a set of features that can be used to predict answer quality, based on an analysis of Mechanical Turk logs. Several studies have shown that it is possible to combine automatic prediction methods (such as Bayesian or generative probabilistic models) with additional input from the crowd to further improve the accuracy of the predictions (dos Reis et al., 2015; Hare et al., 2013; Ipeirotis et al., 2010; Loni et al., 2014; Simpson et al., 2013). Other studies have analysed and compared different algorithms (Zheng et al., 2017; Wang et al., 2015; Sheshadri and
Lease, 2013), emphasizing the need for more research to understand the interplay among different sets of design parameters on the overall performance.

Most of the popular crowdsourcing platforms support aggregating results with simple algorithms, such as Majority voting (MV) which has also been used by many existing studies and projects (Liu et al., 2012; Tran-Thanh et al., 2013; Hung et al., 2015; Yang et al., 2015; Zhang et al., 2017). More sophisticated inference algorithms based on probabilities and statistics were studied and proved to be more efficient in corresponding areas. Probabilistic inference is popular among these sophisticated algorithms and have been investigated by many previous research (such as Bayesian based approaches (Demartini et al., 2012; Kamar et al., 2012b; Ma et al., 2015), EM based approaches (Hung et al., 2015; Whitehill et al., 2009), Bayesian network (Suermondt and Cooper, 1991), Active Learning (Nguyen et al., 2015)). A large portion of these studies also use Bayesian models to predict correct labels: Kamar et al. (2012b) uses a Bayesian model to classify galaxies from Zooniverse project; Venanzi et al. (2014) presents a community-based Bayesian model; Simpson et al. (2013) uses a Bayesian combination of multiple classifiers to classify images from Galaxy Zoo Supernovae; Ramchurn et al. (2015) uses IBCC (independent Bayesian classifier combination) and Gaussian Process. Though Tran-Thanh et al. (2015) has a focus on cost-effective workflows and Demartini et al. (2012) using the probabilistic method to optimize cost, quality and time, the estimation algorithms they use is generally applicable in crowdsourcing data aggregation. Some research focuses on inference in Bayesian belief networks (Suermondt and Cooper, 1991; Murphy and Russell, 2002; Bachrach et al., 2012b), which inspired this research to model the complex crowdsourcing classification problem as a graph, will be elaborated in chapter 4. Expectation Maximisation (EM) (Dawid and Skene, 1979) is another popular algorithm used in many studies and real-world applications. Ipeirotis et al. (2010) leverages EM algorithm to separate unrecoverable error and bias by assigning a score to each crowd worker which accordingly improves the estimation of the correct result. Wang et al. (2013a) proposes a recursive EM algorithm, in which instead of always operating on the whole dataset, it integrates new data in a streaming way which reduces the computation overhead. Similarly, Hung et al. (2015) introduces the increment EM algorithm (also called i-EM) which estimates worker credibility based on the previous iteration to avoid computing on the whole data every time, which has shown better estimation with less computation.

On the other hand, some previous work already has a focus on having an overall understanding of the available algorithms and their performances. Sheshadri and Lease (2013) presents an open source SQUARE framework for benchmarking purpose and provides not only datasets, evaluation metrics but also implementations of some popular methods such as MV, ZenCrowd (Demartini et al., 2012), RY (Raykar et al., 2010) and GLAD that considered worker performance (Whitehill et al., 2009). Zhang et al.
especially look into the performance of current EM-based consensus algorithms and propose an Adaptive Weighted Majority Voting (AWMV) which has shown better performance than the four EM-based algorithms which are empirically evaluated using the same dataset: DS (Dawid and Skene, 1979), ZenCrowd (Demartini et al., 2012), RY (Raykar et al., 2010) and GLAD (Whitehill et al., 2009). Zheng et al. (2017) has a comprehensive review of the existing inference methods in Crowdsourcing, presenting a thorough summary of these methods from task/worker modelling perspective (see Figure 2.3, and empirically evaluated them with 5 representative datasets each of which has a different task type and task size. However, the result indicates there is no single algorithm consistently working well on all datasets and shows choosing an effective inference algorithm remains a challenge for crowdsourcing data aggregation and quality assessment.

<table>
<thead>
<tr>
<th>Method</th>
<th>Task Types</th>
<th>Task Modeling</th>
<th>Worker Modeling</th>
<th>Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>MV</td>
<td>Decision-Making, Single-Choice</td>
<td>No Model</td>
<td>No Model</td>
<td>Direct Computation</td>
</tr>
<tr>
<td>BOC [57]</td>
<td>Decision-Making, Single-Choice</td>
<td>No Model</td>
<td>Confusion Matrix</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>MFD [58]</td>
<td>Decision-Making, Single-Choice, Numeric</td>
<td>No Model</td>
<td>Worker Probability, Confidence</td>
<td>Optimization</td>
</tr>
<tr>
<td>Multi [31]</td>
<td>Decision-Making</td>
<td>Latest Topics</td>
<td>Diverse Skills, Worker Bias, Worker Variance</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>ROS [59]</td>
<td>Decision-Making</td>
<td>No Model</td>
<td>Worker Probability</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>VI-E [33]</td>
<td>Decision-Making</td>
<td>No Model</td>
<td>Confusion Matrix</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>VI-MF [33]</td>
<td>Decision-Making</td>
<td>No Model</td>
<td>Confusion Matrix</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>LFC, N [41]</td>
<td>Numeric</td>
<td>No Model</td>
<td>Worker Variance</td>
<td>Probabilistic Graphical Model</td>
</tr>
<tr>
<td>Mean</td>
<td>Numeric</td>
<td>No Model</td>
<td>No Model</td>
<td>Direct Computation</td>
</tr>
<tr>
<td>Median</td>
<td>Numeric</td>
<td>No Model</td>
<td>No Model</td>
<td>Direct Computation</td>
</tr>
</tbody>
</table>
workflow for generating taxonomies, as well as inference methods to deduce the parent-child relationship, while Otani et al. (2016) focuses on the task where a parent-child relationship exists between two adjacent classification steps, and propose label aggregation methods that adapt from the existing Generative Model of Labels, Abilities, and Difficulties (GLAD) method (Whitehill et al. (2009)) by considering the hierarchical class-subclass structure. In addition, Wu et al. (2012) investigates the sequential data labelling scenario and present Sembler to ensemble crowd sequential labellings by leveraging the statistical correlation and dependency among multiple instances/sentences which is domain specific and not applicable to other multiple-step classification where no such statistics can be exploited. Parameswaran et al. (2011) and Kamar and Horvitz (2015) particularly looks at the multiple-step image classification tasks while both took the approaches that are not easy to be generalised to suit for other multiple-step classification. Parameswaran et al. (2011) explicitly formulates the classification task as a human-assisted graph search problem, presenting the dimensions characterising the different types of classification and developing algorithms to optimise the questions to be asked (at the different node) which is evaluated with simulation. On the other hand, Kamar and Horvitz (2015) focuses on optimising worker allocation in the hierarchical classification task (HCT) and develop answer models and evidence models for HCT consensus while both models are constructed with supervised learning, assisting with the Sloan Digital Sky Survey (SDSS) features identified by machine visions available for GalaxyZoo dataset.

There is also research dedicated to automatic hierarchical classification where a taxonomy is given and a parent-child relationship among classes exists, but all are bound to a certain domain. For instance, Dumais (2000) investigates automatic hierarchical classification using Support Vector Machine with existing web pages whose categories are known as training data. Su et al. (2006) presents an automatic method to classify structured web databases by leveraging probing queries, the returned count of query result and the SVM classifier. Such automatic hierarchical classification not only needs existing labelled data as training data but also focus on the classification where answers to further classification steps down the line (child classes) are always a sufficient condition to confirm the answer to the previous classification step (parent classes).
<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Who use/study it</th>
<th>Why (Pros and Cons)</th>
</tr>
</thead>
</table>
| Majority Voting (MV) based     | Liu et al. (2012)  
Tran-Thanh et al. (2013)  
Hung et al. (2015)  
Ho et al. (2013)  
Yue et al. (2014)  
|                                |                                                                                 | Cons: simple majority voting does not consider item and worker performance difference; weighted voting considers worker performance to a certain degree, but it can be challenging to determine the proper weights. |
| Bayesian Model based           | Welinder et al. (2010)  
Demartini et al. (2012)  
Kamar et al. (2012b)  
Simpson et al. (2013)  
Ramchurn et al. (2015)  
Ma et al. (2015)  | Pros: good for reasoning under uncertainty, can capture uncertainty of worker’s expertise.                                                   |
|                                |                                                                                 | Cons: can be computationally challenging; inadequate/improper model assumptions can bias the estimates.                                           |
| Expectation Maximisation (EM)  | Dawid and Skene (1979)  
Whitehill et al. (2009)  
Raykar et al. (2010)  
Hung et al. (2015)  | Pros: use the confusion matrices to capture the workers performance.                                                                           |
| based                          |                                                                                 | Cons: do not particularly consider bias, do not consider uncertainty                                                                         |
| Active (Deep) Learning         | Costa et al. (2011)  
Nguyen et al. (2015)  | Pros: semi-supervised learning, needs less data compared with randomly selected samples as training data.                                     |
|                                |                                                                                 | Cons: prone to bias (selection criteria, identified pattern)                                                                                  |
| Message Passing                | Karger et al. (2011)  
Ross et al. (2011)  
Ho et al. (2013)  | Pros: represent worker’s reliability with 0 and 1, iterative process to update worker message and task message, can apply to graphical model. |
|                                |                                                                                 | Cons: poor performance when the data is sparse (particularly when there are a large number of categories and workers); message update can be time-consuming and memory-intensive. |

*Table 2.2: Popular Inference algorithms for aggregating labels from multiple workers*
Chapter 3
Methodology

This chapter explains the methodology of this research. It starts with the foundations of multiple-step classification (3.1) which shows the typical classification scenario in the real world and defines the range of scenarios this research works over. Terms and notations used across this research are then defined. It follows with how the corresponding classification is normally done in an abstract way, envisions the ideal components to complete such a classification scenario, and sets forth the two models this research will develop to explore the classification space tackling the research questions. It then explains the two main models, Aggregation Model (3.5) and Classification Model (3.6), and how they are going to be used respectively and together (3.7).

3.1 Foundations of Multiple-Step Classification

This study focuses on a specific case where more than one human classification step is required to complete a classification (multiple-step classification) task. This section elaborates the multiple-step classification, which essentially is a specific type of workflow in itself (with only humans involved) or can be part of other workflows (e.g. a workflow with both the automated process and the human process involved).

Looking at existing entity classification or image classification projects, no matter how complicated the pre-defined categories might be, there is one thing in common: the
pre-defined categories to which the given entity/object need to be classified into are “structured” (either there is hierarchical parent-child relationships, or interdependent relationships between categories). In the entity classification case, users are either asked to pick a category from a few given categories (eg. “Person”, “Place”) or classify based on an existing ontology containing all the classes (aka, categories). In both cases, the categories can be seen as “structured”. In simple image classification, such as identifying whether an image contains a car or not (binary options in Figure 3.1) or classifying an image into a few categories (multiple-options in Figure 3.2), it can be treated as a one-level structure. In complicated classification, such as the example from GalaxyZoo project\textsuperscript{14}, the pre-defined categories contain more levels and each level has various number of options. The cases this research handles are complicated tasks with \textbf{multiple steps} (levels) and/or \textbf{multiple options} (categories).

Taking a close look at existing classification tasks from Zooniverse, a large percentage of these tasks are multiple-step tasks, as shown in Figure 3.3. Each of these projects uses a slightly different type of workflow to classify an object, for example, an image, according to a number of criteria. The workflow consists of several steps, typically in the form of a multiple-options question to be answered by the crowd. Sometimes there are dependencies between steps as the answer chosen for one question prompts other questions to be displayed (e.g., Figure 3.4). Other times, workflows are sequences of independent, though related questions (e.g., Figure 3.5). This research calls this
multiple-step classification to distinguish it from the general definition of workflow (see table 3.1)

The multiple-steps classification can be modelled as a graph, where a node corresponds to a classification option. Each node can be reached via multiple paths from the root, which prompts the first question of the workflow. Each step in the multiple-steps classification is associated with a Question to classify an object according to a criterion, which is rendered on the screen as text (e.g., “Is this animal a zebra, a gazelle, etc.?”) and/or images (e.g., “Does this galaxy look like this triangle or like this rectangle?”). To answer, the crowd needs to choose one of the different options; for example, the question about the animal type in Snapshot Serengeti has 54 possible answers. Once an option has been selected, the classification advances to the next step with a new question. For example, in Snapshot Serengeti, no matter what animal type was chosen, the user is asked to count how many animals of that type they can see (Figure 3.5). In DarkSkies the dependencies between questions are stronger. Once the user has decided whether the nocturnal image shows a city, stars in the skies, an astronaut, etc. there are two follow-up independent questions: one related to how cloudy the image is (with three options: “cloudy”, “some clouds”, “clear”, see Figure 3.4) and a second one about the quality of the image (“sharp” or “blurry”). GalaxyZoo has more complex steps in which questions vary based on what has been chosen in previous classification steps. For instance, the first question is “Is the galaxy simply smooth and rounded, with no sign of a disk?” and three options are provided: “Smooth”, “Features or disk”, and “Star or artifact”. When choosing “Smooth”, a new question will be asked “How rounded is it?” and available options are “Completely round”, “In between” and “Cigar shaped”. In case “Features or disk” is chosen as the answer to the first question, a different set of subsequent questions

---

In most cases, the workflows are tree-shaped, though there are some instances on the Zooniverse platform where this was not the case.
will be asked as indicated in the GalaxyZoo decision tree (Figure 3.6). Eventually, if the image is not a “star or artifact”, the last or second last question will be prompted asking the crowd to classify “Is there anything odd?” and classify the odd feature if applicable. Later chapters (4 and 6) elaborate the related model this research proposes for such multiple-step classification cases and its applications.

3.2 Definitions

Before presenting the methodology of this research, it needs to be clear about some of the key terms employed by this research. Table 3.1 presents the basic terms and definitions that will be used throughout this work.
Formal definitions, notations that are used to represent the multiple-step classification, and notations to elaborate individual inference algorithms in the proposed models, are defined in Chapter 4.

### 3.3 Overview

A typical classification scenario includes following major steps: taking the objects that need to be classified as input; executing the tasks and workflows to elicit classifications; after classification data is collected, run the algorithms to aggregate the collected classification and determine the true label for each object being classified, such as the one shown in Figure 3.7. The classification tasks can have multiple steps, with each step having a different number of options, and the workflows executed can have a number of
Table 3.1: Definitions

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task</td>
<td>a general term referring to an action or a series of actions need to be executed by a crowd worker.</td>
</tr>
<tr>
<td>Classification Task</td>
<td>task classifying objects into given categories, it could be a simple task (one question) or a relatively complex task (more than one question).</td>
</tr>
<tr>
<td>Microtask</td>
<td>a task is decomposed into smaller units making it easier for the crowd. One microtask is equivalent to one question in a classification task.</td>
</tr>
<tr>
<td>Workflow</td>
<td>microtasks are arranged/chained in a way to complete the task. In this thesis, we define three types of workflows for classification based on how the initial classification options are produced: auto, hybrid and naive. This is fully investigated in Chapter 5.</td>
</tr>
<tr>
<td>Multiple-step classification</td>
<td>a multiple-step classification task is a classification task that has been decomposed into microtasks and is executed by the crowd in multiple steps (posing as multiple sequenced questions, as defined in this study in section 3.1). A multiple-step classification task in this definition is a specific form of a workflow that involves only humans in multiple classification steps.</td>
</tr>
<tr>
<td>Question</td>
<td>classification task asked of the user to elicit/assign a label to an attribute of the object to be classified.</td>
</tr>
<tr>
<td>Options</td>
<td>the set of possible labels (referring to the corresponding category)</td>
</tr>
<tr>
<td>Chosen Option</td>
<td>the option a user chooses per question (can be also referred to as the label/category user selected, which in the entity classification context, also known as the annotation)</td>
</tr>
<tr>
<td>Correct Label</td>
<td>the correct label for a question</td>
</tr>
<tr>
<td>Chosen Path</td>
<td>the user chooses a set of labels for entire workflow based on</td>
</tr>
<tr>
<td>Correct Path</td>
<td>the correct set of labels for entire workflow</td>
</tr>
<tr>
<td>Workflow Graph</td>
<td>the multiple-step classification (workflow) can be modelled as a directed acyclic graph (DAG), in which the root node represents the object under consideration and all other nodes are classification options. The model is elaborated in Chapter 4.</td>
</tr>
<tr>
<td>Node</td>
<td>representation of an option in the proposed model</td>
</tr>
<tr>
<td>Node Level</td>
<td>the sequence that the question is presented to the user within a workflow</td>
</tr>
</tbody>
</table>

Figure 3.7: Normal Flow for Classification Tasks

steps involving human, machine or both (hybrid-mode). These are different parameters to consider in crowdsourcing contexts. This research will explore these aspects by developing necessary components for the classification process in which different settings can be tested and evaluated, and accordingly provide insight into the task, workflow design and the quality of the classification results. In an ideal case, these components then can be integrated into a general classification framework (or system), the vision of which is
depicted in Figure 3.8, and contains three components: the classification component, aggregation component, and quality metric component. Such a system will enable task requesters to design classification task/workflow easily and dynamically determine the proper inference algorithm to use based on characters of task and workflows, data collected, and the desired metrics. This research concentrates mainly on the classification and aggregation component, in order to identify some of the important factors affecting the classification accuracy.

Most previous studies around crowdsourcing workflows have focused on the design of the workflows and have shown that a particular type of workflow can be crowdsourced effectively (in terms of the accuracy of outputs, the budget, time, etc) (Little et al., 2009; Bernstein et al., 2010; Tran-Thanh et al., 2015). In some cases, researchers have proposed bespoke quality assurance methods for their workflows (Lintott et al., 2011; Willett et al., 2013). In this systematic study in crowdsourcing classification tasks, we need a model in place to allow us to explore a range of various workflows available for the classification task domain. Since this research focuses on the quality aspects of the classification result, it is important to understand how quality assessment can be done in a multiple-steps classification context. Though there are many inference algorithms as mentioned in section 2.4.3, it is not clear which algorithm or what aggregation strategy is suitable for inferring correct answers in multiple-step classification contexts, which is another focus of this work. Study via the classification model and the aggregation model, allows us to understand better the important factors that could affect the classification quality and to further validate these specific factors eventually in a controlled environment so that in the end, specific design principles for classification tasks could be summarised and recommended to the task requesters.
To address the research questions concerning the relationship between task, workflow design and quality of answers in classification, this work employs the case study research method (Sokolovsky, 1996; Gillham, 2000) combined with experimental research (Kaplan et al., 1988; Ross and Morrison, 2003) to explore various task design and empirically evaluate its performance measured by the quality of classification result. This work focuses on two types of classifications as stated in Section 2.2 of Chapter 2: entity classification and image classification. DBpedia entity classification and Citizen Science image classification projects are used to study the corresponding crowdsourcing classification tasks. In order to tackle the two RQs presented in section 1.2, this work investigates two models in classification and aggregation components targeting RQ2 and RQ1 respectively. Later, it will have the important factors tested and validated by combining both classification and aggregation components (further insights into RQ2). Implications for classification task design will be drawn from these and corresponding aggregation strategies are recommended.

- **RQ1**: How to assess the quality of a classification task with multiple steps (questions)?
  To answer this question, a method for evaluating the accuracy of classification results in multiple-step classification needs to be devised. This corresponds to the aggregation component in the visioned ideal framework (Figure 3.8). This is achieved by comparing the performance of different algorithms (with aggregation strategies) on datasets obtained via different multiple-step tasks. The aggregation model for considering multiple-step classification as a graph and using different inference algorithms and strategies to infer true labels is explained in Section 3.5.

- **RQ2**: Given a fixed number of available categories, how do the different task and workflow design affect the quality of the classification result? When the classification involves a large number of options, whether the multiple-steps design is able to achieve a higher classification accuracy? This research question will help us understand the impact of task and workflow design on classification quality by exploring workflow alternatives when all the available categories are given. This corresponds to the classification component in the visionary framework (Figure 3.8). A classification model is created to allow comparison of different classification workflows, and is evaluated with an entity classification case. The alternative workflow design and how it is applied to the DBpedia entity classification scenario is elaborated in Section 3.6. Upon observing the workflow performance through entity classification case in investigating the first part of RQ2 and performance of different multiple-step classification cases (design) from archived citizen science projects investigated in answering RQ1, the study goes back to one of the existing multiple-step case, implementing the task with a different design and aggregating with the method proposed in this research, and carries out a comparative analysis in order to answer the second part of RQ2 and get further insight on classification
task design. Section 3.7 presents the work to validate the multiple-step design in which both the workflow model and aggregation model proposed in this research are integrated.

3.4 Scenarios for Evaluation of Research Questions

The scenarios this study will look at are large knowledge bases such as DBpedia\(^2\), Wikidata\(^3\) and large-scale crowdsourcing projects, such as citizen science\(^4\) where millions of volunteers providing lots of classifications (Cox et al., 2015). On one hand, for a knowledge base like DBpedia, one of the classification problems is entity classification based on the DBpedia ontology\(^5\) that has many categories (multiple-options) and a parent-child structure exists between categories. The existing work seems unable to solve the problem in an automated way. At the same time, simply presenting all categories to the crowd to choose is not effective (section 2.2.2). On the other hand, most of the citizen science projects use classification tasks to classify features from an object in the form of image, video or audio, putting the object into a few well-defined categories. For instance, to identify the galaxies in GalaxyZoo\(^14\) project, participants are asked to identify the given image is “Smooth”, “Features or disk”, and “Star or artifact”. However, the task normally involves further classification (either dependent or independent of the previous classification) to collect more data that helps to identify more features of the objects under study (as shown in Figure 3.6 in section 3.1 where we define this as Multiple-step Classification).

Experiments will be performed on the chosen projects and evaluated based on gold standards: (1) explore multiple projects with different multiple-step classification task design, applying aggregation model to infer the correct answers which will be compared against gold answers, then derive implications on how aggregation should be done in multiple-step classification and how these factors affect the quality of classification results. (2) instantiate classification model to investigate alternative task designs for a specific project (also multiple-step classification) and assess the performance of each design (task/workflow). (3) validate the findings obtained above in a chosen classification project where non-mandatory multiple-step design can be improved to achieve higher answer accuracy.

Conceptualisation for the two models this study covers is elaborated in section 3.5 and 3.6: Model for aggregation (in multiple-step classification) can facilitate in inferring the

\(^2\)http://wiki.dbpedia.org/
\(^3\)https://www.wikidata.org/wiki/Wikidata:Main_Page
\(^4\)http://www.citizensciencealliance.org/projects.html
\(^5\)http://mappings.dbpedia.org/server/ontology/classes/
correct answer, hence help to evaluate the quality of classification in multiple-step classification cases; Model for classification workflows allows us to test different classification task designs.

### 3.5 Aggregation in Multiple-step Classification

The goal is to infer the correct label for the Multiple-step classification defined in 3.1. During the process, three citizen science projects are chosen, and a graph model is conceptualised. This study proposes an aggregation architecture that leverages different existing algorithms and aggregates the result with strategies that can produce a true label path (within the graph) with high accuracy. The aggregation model is a proxy needed to assess the multiple-step classification result. The overall aggregation process can be correspondingly represented in Figure 3.9.

**Figure 3.9:** Overall process for answer inference in Multiple-step Classification tasks

The main motive of this aggregation step is to find a good way to aggregate the result for multiple-step classification. On that account, this work aims to evaluate the proposed aggregation strategy implemented with three chosen algorithms to get insight on how to aggregate to achieve high-quality results in multiple-step classification. Evaluation of this aggregation strategy (together with the corresponding algorithms) looks at the classification accuracy, and the potential of leveraging the top \( k \) ranked result by checking precision at a certain rank. Chapter 4 elaborates on the model design, experiments and corresponding evaluations.

### 3.6 Classification Workflow

Based on existing studies (section 2.2) and observations from popular classification projects\(^8\),\(^36\), this work abstracts the classification process as a Classification Workflow Model containing three sequential phases that were called “predictor”, “error detector” and “error corrector” accordingly. This is a reflection of how classification can be done in general, while it does not mean every phase is compulsory, except for the predictor

\(^6\) https://en.wikipedia.org/wiki/Ranking_(information_retrieval)
phase. As shown in Figure 3.10, “predictor” is the phase to predict (or choose) the class (aka. label) for the given object/entity, it can be done with different approaches: (1) using existing automatic tool to predict candidate classes if possible, (2) using the crowd to do initial classification to predict the candidate class list (top chosen categories or other selection criteria), (3) using a hybrid approach to predict a list of candidate categories. Depending on different ways of implementing the predictor, the classification workflow can be called Wauto, Wnaive or Whybrid respectively. In particular, in entity classification and image classification, the automatic predictor and hybrid predictor could be implemented using very different techniques, but the overall three-stage workflow is applicable to both cases. After the predictor phase, comes in the error detector and corrector phase that will fully utilise the wisdom of the crowd, which is inspired by previous research (Bernstein et al., 2010; Little et al., 2010b).

As a starting point, this work takes the entity classification problem in DBpedia and systematically analyses the main dimensions of crowdsourcing exercise with the aim of devising alternative workflow to improve classification accuracy. It utilizes the “structured” nature of ontology and devises a tree-search model with three workflows. Each workflow has a predicting, (optional) error detection and error correction stage while distinguishes itself by using auto, human, and semi-auto (hybrid) predictor respectively. Thereupon, it compares the performance of each workflow by measuring the accuracy, precision and the “cost of efforts” (number of steps to complete a classification). This is part of the study to deal with RQ2 and the contribution is detailed in chapter 5. As section 2.2 indicates all classification tasks have a similar “structured” nature of the available options no matter if it requires a simple step or multiple steps independent/interdependent classification. As a result, this model can be applied to test various workflows in classification tasks that could involve a mandatory predictor step (human, hybrid or auto), an optional error detector and error corrector step which is done by the crowd.

This study will evaluate the performance of proposed workflows (Figure 3.10), namely: \( W_{naive}, W_{auto} \) and \( W_{hybrid} \), respectively based on predictors \( P_{naive}, P_{auto} \) and \( P_{hybrid} \). It evaluates the accuracy of the predictors, and compares the overall precision and number
of steps taken of all three workflows. Chapter 5 demonstrates the detail model and its application in DBpedia entity classification scenario.

3.7 Combining Workflow and Aggregation

![Figure 3.11: Integrate both classification model and aggregation model to validate the important design factors](image)

Based on the insights drawn from experiments in the aggregation model (section 3.5) and classification model (section 3.6), this work then looks into combining both models and validates them in a controlled environment by testing the corresponding variation, whether using multiple-steps for a classification with a large number of options can affect the quality of classification. As shown in Figure 3.11, this work incorporates both models investigated earlier and validates the insights obtained from Chapter 4 and 5 regarding multiple-step classification.

In choosing a task to validate the design factors by integrating both Classification model and Aggregation model in a multiple-step classification, it considers the typical challenge of having a relatively large number of categories/options (a common challenge in both entity classification and image classification case) to classify a given object. The Snapshot Serengeti\(^{15}\) task could be a good candidate for such validation because: First, it is a multiple-step classification task in its original design. Second, it has 54 categories for the species and 12 options for the number (or number range) of the animals. Last but not least, the original design is to use non-mandatory filters as a tool to facilitate the crowd, which makes it possible to explore turning such step into a further multiple-step classification.

The ultimate purpose of this integration work is to validate the multiple-step design insights drawn from both models. As a result, the evaluation will mainly involve comparing the accuracy of classification between the dataset gathered by the original design and the dataset obtained from the new design. Classification accuracy will be assessed by comparing the result produced via the aggregation model to the gold standards, to see whether the new design does improve the classification accuracy. It will also compare the classification accuracy for the step which did not further include design variations,
such as the classification step on the number of animals, to understand how the subsequent steps are affected (or not affected) by the changes of preceding step(s). Details of this work are elaborated in Chapter 6.
Chapter 4

Aggregation in Multiple-step Classification

This work focuses on conceptualising multiple-step classification cases and investigates the corresponding aggregation in such cases. The experiments take existing classification tasks, collect crowd labels and aggregate the result with the proposed strategy. It chooses three popular algorithms and applies them to different datasets generated by different tasks, in order to compare the performance of these aggregation methods and find out whether the proposed aggregation strategy helps to improve the classification accuracy.

4.1 Overview

Previous research has proposed a range of methods to infer and predict the crowd answers (as seen in section 2.4.3). Whilst all methods have their benefits, most of them work on relatively simple task models that consist of a single question with one or more answers (Sheshadri and Lease, 2013; Quoc Viet Hung et al., 2013; Zheng et al., 2017), with a few focusing on specific hierarchical classification where a parent-child relationship exists between classification steps (Dumais, 2000; Bragg et al., 2013; Otani et al., 2016). The scenario this work deals with is different. Consider the example in Figure 1.2, which
Chapter 4 Aggregation in Multiple-step Classification

is taken from a citizen science project in which pictures were taken in the Serengeti national park in Tanzania are analysed online by thousands of volunteers. The crowd is asked to answer a series of related, yet independent questions about what they see in the image, including the types and number of animals.

The aggregation model proposed in this work (4.2.3) is applicable to tasks such as the Snapshot Serengeti one that spans over several multiple-choice questions, that need to be answered by the crowd. As such a task only involves human input, it is considered as a specific type of workflow (naive workflow) in our study. Though a few previous works tried to address similar multiple-step classification (see section 2.4.3), they are either limiting it to the hierarchical classification scenarios where a parent-child relationship exists between classification steps (Dumais, 2000; Bragg et al., 2013; Su et al., 2006; Otani et al., 2016) or restricting the method by having to involve additional information (Parameswaran et al., 2011; Kim et al., 2002a; Wu et al., 2012; Kamar and Horvitz, 2015) such as the machine identified features of the image or frequency/correlation among word usage. Consequently, these existing approaches are not easily applicable to other multiple-step classification cases. There is a need to have a more generalised and intuitive approach. This research will investigate an answer inference approach that can handle multi-steps classification. This work proposes using the graph to model a microtask crowdsourcing workflow (section 4.2.1) and to support inference algorithms in making decisions about correct labels for classification tasks with multiple-questions, where the answer to one question does not have to be the sufficient condition to or imply the answer to the previous question is correct. It uses three inference algorithms, majority voting (Paulheim and Bizer, 2014; Quoc Viet Hung et al., 2013), message passing (Karger et al., 2011) and expectation maximisation (Dawid and Skene, 1979; Whitehill et al., 2009), which have been used in quality assurance in microtask crowdsourcing before. Based on the aggregation model proposed in section 4.2.3, this study implements variations of these algorithms that work on graph-like microtask crowdsourcing workflows with multiple steps (as opposed to single multiple-choice questions that are shaped as trees with a question root and several answer leaves) to determine the correct answer paths for each set of related questions in the workflow. It evaluates the performance of the proposed dependency – aware adapted approach, the existing naive approach, and the run-on-path approach where a full labelling path is considered as an atomic, upon individual algorithms for overall accuracy and a more refined measure which looks at accuracy in individual node level of the workflow graph.

4.2 Aggregation in Multiple-step Classification

In this section, we first create a model that treats answer inference as a path search problem in a graph with questions and answer options as nodes for multiple-step classification tasks consisting of several independent or dependent questions like the one
presented in Figure 1.2, Figure 3.4, Figure 3.5 and Figure 3.6. Then it shows how each chosen algorithm in this study works individually, which starts with established algorithms that have so far been applied to simpler classification problems and explains in detail how they work using notation from the proposed aggregation model. The model details and different aggregation strategies to compute the most likely correct answer path in the graph are presented in Section 3.5. Finally, it introduces the evaluation framework (quality assessment metric) to assess the aggregation strategy presented in Figure 4.1.

In the context of the workflow graph, there are two basic ways to run each algorithm: (i) a baseline approach, referred to as run-on-path, which assumes that individual algorithm(s) (such as MV, MP, and EM) can be directly applied on labeled paths; and (ii) a node-level based approach in which individual algorithm(s) treats answers at different levels in the workflow independently, and then assemble the label path. The second approach is relevant to the algorithms that take into account the performance of the crowd in their computations. By studying it, it allows those sophisticated algorithms to be able to better identify those labellers who, while not doing so well overall, are very skilled at a particular step (sub-task) in the multiple-step classification (workflow). At the same time, the second approach can have two different assembling strategies, this research refers to them as naive and dependency-aware adapted respectively. Figure 4.1 illustrates the detailed process.

### 4.2.1 Problem Formalisation

A classification task which only has one single question and a few options to choose from, such as the one shown in Figure 3.1 and 3.2, is generally considered as a simple task (with simple workflow). It looks like a simple tree structure where the classification starts with a root node which refers to the object to be classified and has a few branches which represent the available options. In a similar way, a relatively complex task that involves either a large number of options for one question, or more than one question and more options, will be more like a tree with branches which have further branches and leaves. If we draw such a ‘tree’ for the three multiple-step classification tasks shown in section 3.1, we can see each of them uses a different type of workflow consisting of several independent/interdependent steps. Each step in the workflow is associated with a Question to classify an object according to a criterion. To answer the question the crowd needs to choose among a set of Options. Figure 3.4 involves a minimum of one step and a maximum of three steps for the classification task. Figure 3.5 has a fixed two steps to complete a classification task and each step has more than ten options. For the GalaxyZoo\(^1\) task, it can involve a minimum one step and a maximum of nine steps to complete a classification, as shown in Figure 3.6. It is notable though these different

---

\(^1\)https://data.galaxyzoo.org/gz_trees/gz_trees.html
Figure 4.1: Applying different algorithms and aggregation strategies

tasks have various number of questions, and various number of available options, there are indeed nodes which have more than one parent node.

As a result, the multiple-step classification workflow can be modelled as a directed acyclic graph (DAG), where the root node is the object under consideration and all other nodes are classification options. Each node can be reached via multiple paths from the root, which prompts the first question of the workflow.\footnote{In a lot of cases, the workflows are tree-shaped, but some cases are not a tree such as the three tasks presented above.} For a given object $o$, the crowd is asked to carry out a labelling task, which implies answering a series of (independent or dependent) classification questions with a set of labels which identify the outstanding features of the object being classified. A task like this can be defined as a path search problem in a workflow $W_f$ modelled as directed acyclic graph (DAG) with a root entry point and levels (similar to tree levels, representing the number of questions in the task), each corresponding to a set of options as depicted in Figure 4.2. Each node in such a graph represents a particular labelling option. The labelling finishes when a leaf in the graph is reached, that is a label that does not lead to any further questions. In the definitions of this work, the level corresponds to classification question(s) and the level of a node is serialised and counted at the lowest level. It uses level exchangeably with depth of a node which is indicated by the number of edges from the node to the root.
Figure 4.2: Graph representation of an example classification workflow $W_f$ vs. corresponding classic way of looking at the classification with multiple questions.

A directed edge represents a label chosen for the corresponding question related to that node level.

The key definitions are:

**Definition 4.1 (Object).** Represented by $o$, an object in the classification context refers to anything that is to be classified, it could be an entity in the entity classification case or an image for image classification case.

**Definition 4.2 (Multiple-step classification).** Represented by $W$, a multiple-step classification refers to a classification task that has been decomposed into microtasks and is executed by the crowd in multiple steps (posing as multiple sequenced questions, as defined in this study in section 3.1).

**Definition 4.3 (Option).** An option is a label (aka, category) to classify the given object. All available options are represented by $A$ and $a_j$ represents an individual option, where $j \in \{1, ..., |A|\}$. In the context of an entity classification with a given ontology, $A$ corresponds to the ontology.

**Definition 4.4 (Chosen Option).** Represented by $l_o^u$, a chosen option refers to an option user chooses for the classification question being asked. It can be also referred as the label/category user selected, which in the entity classification context, also commonly known as the annotation\(^3\). $L_o$ represents the set of all labels from the crowd for object $o$ and $L^u$ represents the set of all labels from user $u$.

\(^3\)[https://prodi.gy/docs/workflow-first-steps]
\(^4\)[https://gate.ac.uk/sale/tao/splitch21.html]
Chapter 4 Aggregation in Multiple-step Classification

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( o )</td>
<td>the current object being classified</td>
</tr>
<tr>
<td>( O )</td>
<td>the set of all objects in a dataset</td>
</tr>
<tr>
<td>( A )</td>
<td>all available options</td>
</tr>
<tr>
<td>( a_j )</td>
<td>an individual option, where ( j \in {1,...,</td>
</tr>
<tr>
<td>( u )</td>
<td>user ( u )</td>
</tr>
<tr>
<td>( U )</td>
<td>the set of all users who contributed to the current dataset</td>
</tr>
<tr>
<td>( U_o )</td>
<td>all users who have classified object ( o )</td>
</tr>
<tr>
<td>( L )</td>
<td>all labels received from the crowd, and ( L \subseteq A )</td>
</tr>
<tr>
<td>( L_o )</td>
<td>the set of all labels from the crowd for object ( o )</td>
</tr>
<tr>
<td>( L^u )</td>
<td>the set of all labels from user ( u )</td>
</tr>
<tr>
<td>( l^u_o )</td>
<td>the label for object ( o ) from user ( u )</td>
</tr>
<tr>
<td>( l_o )</td>
<td>the inferred label for object ( o )</td>
</tr>
<tr>
<td>( l_{gold_o} )</td>
<td>true label for object ( o )</td>
</tr>
<tr>
<td>( W )</td>
<td>the multiple-step classification representing a specific type of workflow ( W ) which encompasses multiple-steps need to be completed by the crowd</td>
</tr>
</tbody>
</table>

Table 4.1: Summary of key notations

The key notations used can be found in table 4.1. On top of the terms this study defined in section 3.1 and table 3.1, the multiple-step classification (workflow) can be modeled as a directed acyclic graph (DAG) (Figure 4.2), in which the root node represents the object under consideration and all other nodes are classification options. Notations that are specific to the workflow graph model are also defined in table 4.2.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( W_f )</td>
<td>represents the graph based on the workflow of classifying object ( o ), it has node levels to indicate the questions to classify the corresponding attributes of the given object, and nodes to represent the options available for each attribute</td>
</tr>
<tr>
<td>( A_{(n)} )</td>
<td>represents the available options at node level ( n )</td>
</tr>
<tr>
<td>( a_{n(j)} )</td>
<td>represents the individual option at node level ( n ), where ( j \in {1,...,</td>
</tr>
<tr>
<td>( l^u_{o(n)} )</td>
<td>represents the label chosen by user ( u ) at node level ( n ) for object ( o ). Thus, the labelling result ((l^1_{o(1)},l^1_{o(2)},...,l^1_{o(n)})) will represent the ordered list of nodes (the traversal path) visited by user 1 when classifying ( o ), which is called as a label path</td>
</tr>
<tr>
<td>( L^u_o )</td>
<td>the label path chosen by user ( u ) for object ( o )</td>
</tr>
<tr>
<td>( L_{o(n)} )</td>
<td>all labels for object ( o ) at node level ( n )</td>
</tr>
<tr>
<td>( L_{o(n)}(\text{unique}) )</td>
<td>unique labels for object ( o ) at node level ( n ), ( L_{o(n)}(\text{unique}) \subseteq A_{(n)} )</td>
</tr>
<tr>
<td>( L_o )</td>
<td>represents the inferred label path for object ( o ). It is a set of inferred labels for each node level described as ((\tilde{l}<em>{o(1)},...,\tilde{l}</em>{o(n)}))</td>
</tr>
<tr>
<td>( L_{gold_o} )</td>
<td>true label path for object ( o )</td>
</tr>
</tbody>
</table>

Table 4.2: Notations specific to Multiple-step Classification

The problem this study is solving can be defined formally as follows:
Definition 4.2.1. The Correct Labelling Problem:
Given a particular object \( o \), a multiple-step classification based workflow graph \( W_f \), a set of labels \( L_o \) for object \( o \), and (optionally) a set of previous labels from all users on all objects \( L \), the aim is to infer the correct label path \( \hat{L}_o \) in \( W_f \) for object \( o \).

4.2.2 Existing Algorithms

This section presents three representative existing algorithms for aggregating labels: \( MV \), \( MP \) and \( EM \), that have been chosen for inferring the true label from the crowd collected classifications. These are the foundations to understand the proposed adapted dependency-aware approach for inferring true labels in such multiple-step classification. Corresponding details on how each of these algorithms are demonstrated in section 4.2.2.1, 4.2.2.2, and 4.2.2.3 using the notation defined in table 4.1.

Majority voting (\( MV \)) Due to its simplicity, Majority Voting has been used in many microtask projects (Hung et al., 2015; Liu et al., 2012) and is the standard aggregation method in some existing crowdsourcing platforms\(^5\). Given the list of options for a labelling task and an object, the \( MV \) algorithm chooses those options with the highest number of votes from the crowd.

Messaging passing (\( MP \)) This algorithm (Karger et al., 2011) takes into account both the labels and the performance of the labellers. Given an object and a label, it returns \( true \) in case the label is valid for the object and \( false \) otherwise. \( MP \) constructs object and labeller-specific messages to represent the reliability of the particular labeller, and iteratively updates the object and the labeller messages. More specifically, at each object update, it adds up more weight to labels that come from more trustworthy parts of the crowd; and at each labeller update, it adds more trust (a confidence value) to the labeller if the labels they give for other objects are in line with the current estimates of object labels. The iterative updates continue until the algorithm converges or a specified threshold is hit. The threshold for the stopping condition is a parameter that has to be empirically determined. Whilst providing accurate estimations, \( MP \) is also known for its high computational costs as the number of labels and users increase.

Expectation maximisation (\( EM \)) This algorithm considers the performance of the labellers. As a simplified alternative, the \( EM \) algorithm (Dawid and Skene, 1979) involves two steps to infer the true label for a given object. In the first step, the true label for the current object is estimated using simple majority voting, where the input of all labellers is considered equally. Then, in the next step, the error rate of each labeller is estimated based on this result and used in turn to calculate the new estimation for the first step. The steps are alternating iteratively until the algorithm converges and a maximum is found.

Chapter 4 Aggregation in Multiple-step Classification

4.2.2.1 Majority Voting (MV)

It takes as input an object $o$ and the crowd labels $L_o$ and outputs the resulting candidate label $\hat{l}_o$ that received the most votes from the users.

\textbf{Algorithm 1} Majority voting (MV)

\begin{algorithm}
\begin{algorithmic}[1]
\Procedure{findUniqueLabel}{$L_o$}
\State $L_{unique} \leftarrow \{l^u_o\}$, where $L_{unique} \subseteq A$ and $l^u_o \in A$ and $u \leq U_o$.
\State $\hat{l}_o = \emptyset$.
\State $cnt_{max} = 0$; \Comment{this is used to track the number of the most voted option.}
\For{$l \in L_{unique}$} \Comment{for each unique label, check how many votes it gets.}
\If{$\text{count}(l) \geq cnt_{max}$}
\State $cnt_{max} \leftarrow \text{count}(l)$;
\State $\hat{l}_o \leftarrow l$;
\EndIf
\EndFor
\State \Return $\hat{l}_o$;
\EndProcedure
\end{algorithmic}
\end{algorithm}

4.2.2.2 Expectation Maximisation (EM)

Expectation Maximisation (EM) is another algorithm that has been widely used and involves two steps to infer the true label for a given object. In the first step, the true label for the current object is estimated using simple majority voting, where the input of all users is considered equally. Then, in the next step, the error rate of each user is estimated based on this result and used in turn to calculate the new estimation for the first step. The steps are alternating iteratively until the algorithm converges and a maximum is found. It takes as input an object $o$ and all labels $L$. It starts by estimating the true label for each object and each user’s error rate by comparing their answers (using an indicator function $I()$ to check whether the user classifies object to a certain category/class) for all objects they have looked at. The error rate is used subsequently to update the confusion matrix for each user. The output is candidate labels for $o$ with the probability (indicated by $p$) of the corresponding candidate label to be correct.

4.2.2.3 Message Passing (MP)

Message Passing (MP) is an algorithm that takes into account both the labels and the performance of the users. MP constructs object and user-specific messages to represent the reliability of the particular user, and iteratively updates the object and the user messages. More specifically, at each object update, it adds up more weight to labels that come from more trustworthy parts of the crowd; and at each user update, it adds more trust (a confidence value) to the user if the labels they give for other objects are in line with the current estimates of object labels. The iterative updates continue until the algorithm converges or a specified threshold is hit. The threshold for the stopping
Chapter 4 Aggregation in Multiple-step Classification

Algorithm 2 Expectation maximisation (EM)

1: procedure INITIALISE($p_l$)
2: \[ p_l \leftarrow \text{count}(l) \div |L_o| \quad \triangleright \text{probability of } l \text{ being the true label for object } o \ (l \in A) \]
3: while not converged do
4: \[ \theta_l^u \leftarrow \lambda_l^u + \sum_{o \in L_o} p_l \times I(l_o^u = l^-) \quad \triangleright \text{object of label } l \text{ being classified as } l^- \]
5: Estimate confusion matrix:
6: \[ e_{lm}^u \leftarrow \theta_{lm}^u \div \sum_q \theta_{lq} \quad \triangleright q^u \text{ is the accuracy of user } u \]
7: Estimate class priors:
8: \[ pr_l \leftarrow \sum_o p_o l \div |O| \]
9: Calculate class probability for object $o$:
10: \[ p_l \leftarrow p_l \prod_{u \in U} y_{l \rightarrow o}^u \times \prod_m (e_{lm}^u) I(l^u = m) \div \sum_q pr_q \prod_m (e_{qm}^u) I(l^m = m) \]
11: \[ \tilde{l}_o = \text{"\_"}; \]
12: \[ p_{\text{max}} = 0; \]
13: for $l \in A$ do
14: \[ \text{if } p_l \geq p_{\text{max}} \text{ then} \]
15: \[ p_{\text{max}} \leftarrow p_l; \]
16: \[ \tilde{l}_o \leftarrow l; \]
17: return $\tilde{l}_o$;

condition is a parameter that has to be empirically determined. It takes as input an object $o$, a label $a \in A$, all labels received from the crowd $L$ and a threshold $k_{\text{max}}$. $MP$ computes the object message by firstly iterating all previous labels from the users who have been assigned the object $o$ and then looking at whether each label is the same as the given one. In a next step, it uses the object message $x_{a \rightarrow u} (\in L)$ to update the user message $y_{u \rightarrow o} (\in L)$, which is computed by iterating over the labels they have submitted. Until convergence, the object message for object $o$ is aggregated by weighing the user messages (confidence) for that object and the computed sign is stored in $E_{ou}$. $MP$ outputs the candidate label $l$ for $o$ and the sign of whether the label applies or not.

Algorithm 3 Message passing (MP)

1: procedure INITIALISATION($y_{u \rightarrow o}$)
2: for $(o, u) \in L$ do
3: initialise $y_{u \rightarrow o} \sim \mathcal{N}(-1, 1)$
4: initialise $k_{\text{max}}$;
5: procedure ITERATION($k_{\text{max}}$)
6: for $k \in \{1, \ldots, k_{\text{max}}\}$ do
7: for $(o, u) \in L$ do
8: \[ x_{o \rightarrow u}^k \leftarrow \sum_{u^- \in U} E_{ou^-} \times y_{u^- \rightarrow o}^{k-1} (u^- \neq u) \]
9: for $(o, u) \in L$ do
10: \[ y_{u \rightarrow o}^k \leftarrow \sum_{o^- \in U} E_{o^- u} \times x_{o^- \rightarrow o}^k (o^- \neq o) \]
11: \[ x_o \leftarrow \sum_{u \in U} E_{ou} \times y_{u \rightarrow o}^{k_{\text{max}}-1} \]
12: if sign($x_o$) == 1 then
13: \[ \tilde{l}_o = x_o; \]
14: return $\tilde{l}_o$;
4.2.3 Multiple-step Classification Aggregator

This section elaborates the aggregation model proposed in Figure 4.1. The input consists of the object to be examined along with the sets of labels that each algorithm requires. The output is a ranked list of label paths, where a label corresponds to a classification option for one of the questions in the workflow. In general, each algorithm could produce paths that contain nodes unvisited by some of the labellers. For example, in GalaxyZoo, some volunteers might correctly identify the “disk” in the picture, while others might be more prone to identify odd features, no matter the picture is “smooth”, or “features or disk”. In this work, run-on-path, naive and dependency-aware adapted are all implemented for the three chosen algorithms.

As a baseline, in the run-on-path condition, it considers each label path as an atomic label. It then directly applies the three inference algorithms and ranks the candidate paths. The output is a ranked list of all possible paths in the graph, where the visited paths are ranked higher based on votes or other metrics. Unvisited paths get a score of zero. A simplified case of this is to pick the top inferred path from the ranked list and output it as the true label path. On the other hand, in both the naive and dependency-aware adapted condition, it rewards partially correct answers and applies each of the algorithms at each node level in the graph and compute scores for each individual labels \{a_1^1, a_2^1, \ldots a_l^1\} \ldots \{a_1^n, a_2^n, \ldots a_m^n\}. It then considers each combination between the levels as a path \(a_1, a_2, \ldots a_m\). The major difference between naive and adapted is the way it assembles the candidate paths: naive takes the top inferred results from each node and directly assembles them into candidate paths; adapted takes into account the dependency of the nodes between node levels and assures valid paths.

Theoretically, multiple individual algorithms could be used in this answer inference architecture. This work builds upon 3 popular algorithms (MV, EM, MP), each of which has distinct traits such as considering user reliability and microtask easiness, to estimate correct classification under different multiple-step classification settings. To this end, this work proposes a method which can be applied to systematically determine the correct answers for a whole range of multiple-step classifications, spanning over several steps with independent or dependent multiple-choice questions. In the classic approaches, it does not look at the dependency between node levels hence naively putting inferred result from each node level together does not guarantee a valid result. It is obvious that producing a valid path with possible choices should improve the accuracy of the users’ answers. As such, a basic adaptation of the classic algorithms should show some improvement over multiple level workflows. In the following, such a basic adaptation proposed by this study is shown in Algorithm 4.

**Algorithm 4** Dependency-aware approach

1: procedure Predict_By_NodeLevel(L_o)
num levels = n;

for level ∈ range(n) do

if method == mv then

procedure findUniqueLabel(Lo)

Lunique ← \{l_o^u\}, where Lunique ⊆ A and l_o^u ∈ A and u ≤ U_o;

for l ∈ Lunique do

p_l ← count(l) ÷ |L_l| ⊤ percentage of l being voted as the label for object o;

return LC_n ← \{(l, p_l)\} ⊤ list of candidate labels and their percentage for o;

if method == em then

procedure Initialise(p_l)

p_l ← count(l) ÷ |L_l| ⊤ percentage of l being the true label for object o (l ∈ A);

while not converged do

Estimate error rate for user u:

θ_l^u ← λ_l^u + \sum_{o \in L_o} p_l \times I(l_o^u = l^-)

Estimate confusion matrix:

e_l^u ← θ_l^u ÷ \sum_q θ_l^q ⊤ q is the accuracy of user u

Estimate class priors:

pr_l ← \sum_o p_l^o ÷ |O|

Calculate class probability for object o:

p_l ← pr_l \prod_{u \in U_o} \prod_m (e_l^{am} I(l_o^u = m)) ÷ \sum_q pr_q \prod_m (e_l^{qm} I(l_o^u = m)

return LC_n ← \{(l, p_l)\} ⊤ list of label candidates and corresponding probability for o;

if method == mp then

procedure Initialisation(y_u->o)

for (o, u) ∈ L do

Initialise y_u->o (∼ N(−1, 1));

procedure Iteration(k_max)

for k ∈ \{1, ..., k_max\} do

for (o, u) ∈ L do

x_o^k->u ← \sum_{u^- ∈ U} E_u^- × y_{u^-->o}^{k-1} (u^- ≠ u);

for (o, u) ∈ L do

y_{u->o} ← \sum_{o^- ∈ O} E_o^- × x_o^k->u (o^- ≠ 0);

x_o ← \sum_{u ∈ U} E_u × y_{u->o}^{k\_max-1}

if sign(x_o) == 1 then

LC_o.append((x_o, 1.0))

procedure Assemble_MostPossiblePath(Lo)

num levels = n;
Chapter 4 Aggregation in Multiple-step Classification

The dependency-aware adapted approach first processes the labels at different levels in the workflow independently, then assembles the label path from each node level based on the workflow graph. In the adapted approach, not only it rewards partially correct answers from the crowd by applying each of the algorithms at each node level in the graph and compute scores for each individual labels, but also it considers the valid path when inferring the correct path. This work also specially chooses two algorithms that take into account the performance of the crowd in their computations, EM and MP. The EM algorithm sums up all node probabilities along each path to determine the ranking score. The MP algorithm returns true if that particular label at the node level is relevant or false otherwise. This means that it assigns the score for the candidate paths correspondingly either as 1.0 or 0.0. This work mainly focuses on finding out the top inferred label path.

Furthermore, it is always interesting to see whether by taking strength from different algorithms it could further improve the inference accuracy, such as a combination of inference algorithms, which can be switched on and off based on the number and diversity of choices and answers received from the crowd. A heuristic-based aggregation strategy, which could be based on the intuitions and performance observed from different existing algorithms, could be devised, similar to the ensemble method which has been explored in single question tasks such as text classification (Antunes et al., 2011), handwriting recognition (Burak and Stahovich, 2005), or the well-known random forest technique.

A basic form of such heuristic aggregation can be based on following intuitions: 1. the number of unique classifications of an object (defined by $u$) shows the degree that the crowd workers agree/disagree on the classification where the higher number indicates higher degree of disagreement and normally imply the object is either a bit difficult or ambiguous to be classified. 2. the ratio (defined by $r$) between the unique number
of classifications/answers collected from the crowd and the total number of classifications/judgments also demonstrates how diverse the answers are for the corresponding object and hence similarly. 3. As three-sigma rule (Pukelsheim, 1994) in the empirical sciences suggests that almost all values should lie within three standard deviations of the mean in a normal distribution, and theoretically mean plus one, two or three standard deviation(s) covers 68%, 95% and 99.7% of the data. In the case where majority voting might potentially fail (where workers tend to disagree), the number of unique classification or the ratio of the number of unique to the total number of classification for an object lies within the higher range of the distribution. Then it could utilize the skewness (defined by $s$ below) of the distribution for number of unique ($U \sim N(u_\mu, u_\sigma)$) and ratio (defined by $R \sim N(r_\mu, r_\sigma)$) respectively to heuristically chosen bound where $MV$ can be potentially complemented by other approaches. 4. Considering that $MP$ algorithm also involves estimating worker’s performance for a specific label, the intuition is for the objects with a relatively fewer number of unique answers, the object message is more useful in updating the corresponding worker message as it provides relatively “dense” information on whether a worker agrees or disagrees with others.

$$u_{O_i} = \text{num. unique answers (for object } O_i)$$

$$r_{O_i} = \frac{u_{O_i}}{|L_i|}$$

$$s = \frac{\mu_3}{\mu_2^{3/2}} = \frac{\mu_3}{\sigma^3}$$

For a single dataset: let $MV$, $EM$, $MP$ represent the result from $MV$, $EM$ and $MP$ (the three existing algorithms that were chosen for this research), $s$ is the skewness calculated from the dataset indicated by $D$, $\mu$ and $\sigma$ is mean and standard deviation for the corresponding distributions ($U \sim N(u_\mu, u_\sigma)$ and $R \sim N(r_\mu, r_\sigma)$) of all objects in this dataset $D$. Following the intuitions above, the strategy can be expressed as in algorithm 5.

This work chooses workflows from three citizen science projects to test the proposed aggregation model. All these projects use a $W_{\text{naive}}$ workflow in terms of the classification workflow model (Figure 3.10), however, are distinct to each other in terms of the abstract graph model where each question and answer in the workflow is represented in a DAG graph (Figure 4.2). In particular:

- The Snapshot Serengeti project\textsuperscript{15} has 2 levels of categories corresponding to a fixed two-step classification task: the first question (for the first step) contains 54
Algorithm 5 Heuristic aggregation

Require: MV, EM, MP, D

1: procedure CalculateSkewness(D)  
2:     return $s_r, s_u$ \(\triangleright \) skewness based on the ratio and number of unique answers  
3: if $s_r \leq 0.5$ then 
4:     threshold$_r$,lowerbound $\leftarrow r \mu + 0.5 \times r \sigma$  
5:     threshold$_r$,upperbound $\leftarrow$ threshold$_r$,lowerbound $+ r \sigma$ \(\triangleright \) how the ratio thresholds are defined  
6: else 
7:     threshold$_r$,lowerbound $\leftarrow r \mu + r \sigma$  
8:     threshold$_r$,upperbound $\leftarrow$ threshold$_r$,lowerbound $+ r \sigma$  
9: if $s_u \leq 0.5$ then 
10:    threshold$_u$,lowerbound $\leftarrow u \mu + 0.5 \times u \sigma$  
11:    threshold$_u$,upperbound $\leftarrow$ threshold$_u$,lowerbound $+ u \sigma$ \(\triangleright \) how the upper/lower bound of the number of unique are defined  
12: else 
13:    threshold$_u$,lowerbound $\leftarrow u \mu + u \sigma$  
14:    threshold$_u$,upperbound $\leftarrow$ threshold$_u$,lowerbound $+ u \sigma$  
15: for each $O_i \in \mathcal{O}$ do 
16:    mv_answer $\leftarrow$ MV($O_i$)  
17:    em_answer $\leftarrow$ EM($O_i$)  
18:    mp_answer $\leftarrow$ MP($O_i$)  
19: procedure FindCommonAnswer(mv_answer, em_answer, mp_answer)  
20:     return most_common_answer, most_common_answer.freq  
21: procedure Find($O_i, D$)  
22:     return $r_{O_i}, u_{O_i}$  
23: if most_common_answer.freq $\geq 2$ then 
24:     result $\leftarrow$ most_common_answer  
25: else 
26:     if $r_{O_i} >$ threshold$_r$,upperbound and $u_{O_i} >$ threshold$_u$ then 
27:         result $\leftarrow$ em_answer 
28:     else if $r_{O_i} >$ threshold$_r$,lowerbound and $r_{O_i} \leq$ threshold$_r$,upperbound 
29:         and $u_{O_i} >$ threshold$_u$,lowerbound and $u_{O_i} \leq$ threshold$_u$,upperbound then 
30:         result $\leftarrow$ mp_answer 
31:     else 
32:         result $\leftarrow$ mv_answer

options (categories) about type of animals; the second question contain 11 options about the number of that type of animals identified in the first step (see Figure 3.5).

- The Dark Skies project\(^3\) has a workflow where questions vary based on what has been chosen in the previous classification step. It has 3 levels of categories corresponding to one or three classification steps: the first question contains 7 options to identify the object in the picture taken from the sky; only when “city” is identified from the first step, second question containing 3 options related to how cloudy the image is and third question containing 2 options about the quality of the image will be presented (see Figure 3.4).
The GalaxyZoo project\(^\text{14}\) has a more complex workflow in which questions vary based on the option chosen in previous classification step. It has 9 levels of categories corresponding to one to nine classification steps: the first question contains 3 options to identify whether the galaxy is simply smooth and rounded, with no sign of a disk; If choosing “Smooth”, two fixed questions will be asked subsequently with the different number of options respectively. In case “Features or disk” is selected, a different set of subsequent questions will be asked which can reach up to 9 steps; If the image is not a “star or artifact”, the last or second last question will be prompted asking the crowd to classify “Is there anything odd?” and classify the odd feature if applicable (see Figure 3.6).

These chosen workflows exhibit different complexity and are representational for testing the aggregation model for answer path inference. Snapshot Serengeti represents a workflow with shorter path length, but with a relatively larger number of nodes (options) in each level of the graph. Both DarkSkies and GalaxyZoo have a varied length of the path, while having a relatively smaller number of nodes per level. Comparing DarkSkies and GalaxyZoo, DarkSkies is simple with 3 levels of nodes, the second and third levels both contain a small number of nodes, while GalaxyZoo has a longer path and more nodes per level.

### 4.2.4 Quality Assessment Metric

In the absence of standard benchmarks or methodologies for evaluating the effectiveness of answer inference algorithms for the types of microtask workflows this study considers, it defined the following criteria based on metrics from information retrieval (IR). In IR, the aim is to assess the quality of a list of ranked search results for a set of queries against a gold standard. This study considers each of the three algorithms \(MV\), \(MP\), and \(EM\) as an IR system, each object \(O_i\) as a query, and the ranked list of label paths \(LP_{\text{candid}}\) proposed by each algorithm as the result set for \(O_i\). It compares the result set with the gold standard data, which is a ranked list of paths \(LP_{\text{gold}}\).

To measure the performance of the proposed dependency-aware implementation, it employs two state-of-the-art quality metrics used in IR: **Mean Average R-Precision** and **Averaged Discounted Cumulative Gain**. Though the metrics have different characteristics, both were developed to reflect the average quality of an IR system on the given query and document set. The following briefly describes the metrics and their features in more detail. The performance of each method is also evaluated with the commonly used **Accuracy** metric.

**Mean Average R-Precision (MARP)** It first introduces Average R-Precision (ARP) and then its mean variant, which were used in the evaluation. For an \(O_i\), precision at
rank $k$ is defined as the fraction of correct labels in a sub-list of $top - k$ labels from those labels identified by an algorithm. More formally,

$$precision(O_i, k) = \frac{|LP_{gold_i} \cap sublist(LP_{candid_i},k)|}{|sublist(LP_{candid_i},k)|}$$

This study applies ARP with $k <= |LP_{gold_i}|$ - that is, it expects a perfect algorithm to produce a result set such as $LP_{candid_i} = LP_{gold_i}$). More formally,

$$ARP(O_i) = \sum_{k=1}^{|LP_{gold_i}|} \frac{precision(O_i, k)}{|LP_{gold_i}|}$$

To obtain a more complete picture of the performance of a particular algorithm, it averaged the precision over all queries (all objects in the gold standard set in this case). The closer the value to 1.0, the better the performance. More formally, it defines the MARP for an inference algorithm as:

$$MARP = \sum_{x=1}^{|O|} ARP(O_i)$$

**Averaged DCG/NDCG** Discounted cumulative gain (DCG) measures take additionally into account the relevance scale of the returned results and are commonly used in evaluating ranked results Yin et al. (2016); Cheng et al. (2016). In this case, this means the correct classifications should be also ranked higher by the algorithms and their score penalised otherwise. To be independent of the result set size, this study calculates a normalised DCG at every rank position, formalised as follows:

$$DCG(O_i, k) = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{\log_2(i)}$$

In the formula, $rel_i$ is 1 in case the label at the rank $i$ is relevant, and 0 otherwise. For each rank position, it plots a DCG, averaged over all objects. The faster the obtained curve reaches 1.0 (i.e., all correct labels are found), the better the performance of the algorithm.

**Accuracy** To measure the performance of the proposed aggregation approach, this work employs the Accuracy metric which has been commonly used in classification evaluation in previous work Khattak and Salleb-Aouissi (2011); Kamar et al. (2012a); Sheshadri and Lease (2013); Quoc Viet Hung et al. (2013); Zhang et al. (2017); Zheng et al. (2017). Accuracy is a measure allowing us to understand the percentage of correct answers (inferred by algorithms). The accuracy is defined as the percentage of objects that have been correctly inferred. Higher accuracy indicates better performance.
Chapter 4 Aggregation in Multiple-step Classification

77

Accuracy = \sum_{o}^{\mid O \mid} \text{Bernoulli}(L_{\text{gold}_o} == \tilde{L}_o)

The above equation is by default for calculating the accuracy for the inferred label path. As this work uses the dependency – aware adapted node-level based implementation, it makes sense to also evaluate how accurate the inferred label is on each node level. In such context, $L_{\text{gold}_o}[n]$ represents the ground truth for object $o$ at node level $n$ and $\tilde{L}_o[n]$ represents the inferred true label at node level $n$. Hence the accuracy at node level $n$ for the top answer can be calculated by:

$$Accuracy_{level_n} = \sum_{o}^{\mid O \mid} \text{Bernoulli}(L_{\text{gold}_o}[n] == \tilde{L}_o[n])$$

To understand whether the adapted dependency-aware approach is significantly better, it will also run significant testing\footnote{http://onlinestatbook.com/2/logic_of_hypothesis_testing/significance.html} for all algorithms chosen. It will use a standard 5% significance level. For each dataset, it will randomly select 100 objects and select 50 times. The accuracy for each selection is calculated for $MV, MP$ and $EM$ for both naive and dependency-aware adapted approach. This research will use the function scipy.stats.ttest_ind from Python\footnote{https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ttest_ind.html} to perform the two-sided test for naive and adapted samples in all six cases (three workflows, each has two contexts: volunteer and paid).

As the output in the proposed method is a ranked list, it makes sense to consider the top few labels (eg, top k) when calculating accuracy. If the correct label is within the top k labels, it considers the object as being correctly inferred.

$$Accuracy_{\text{top}k} = \sum_{x=1}^{\mid O \mid} \text{Bernoulli}(LP_{\text{gold}_x} \cap \text{sublist}(LP_{\text{candid}_x}, k))$$

4.3 Experiments and Results

The study firstly evaluates the overall approach on two datasets with different workflows: one from Snapshot Serengeti\footnote{http://www.zoology.ubc.ca/groups/arts/ongoing_projects/snss/}, which is based on an iterative workflow where each iteration consists of several independent questions; and a second one from the Cities at Night project\footnote{http://www.citiesatnight.com/}, which uses microtask crowdsourcing to analyse night-time photographs taken by astronauts onboard the ISS via a workflow in which questions and answers are arranged in a decision tree. Then it also replicates the existing workflows from citizen science classification projects and evaluates the quality in a paid crowdsourcing environment using the proposed aggregation strategy. For the later experiment, it uses a more complex workflow from GalaxyZoo project where several different questions were
asked in sequence depending on the answers to previous questions and can involve a maximum of nine classification questions.

4.3.1 Experiment 1: Initial Investigation of Dependency-aware Aggregation strategy vs. Path as Atomic Label Aggregation

4.3.1.1 Data

For the first part of the experiment, it started with two datasets from Citizen Science projects. The first one is from the Snapshot Serengeti project and consists of all crowd classifications within the time span from December 10th, 2012 until July 17th, 2013: 7,800,896 labels from 890,280 volunteers for a total of 66,892 objects. For the evaluation, it used a gold standard with curated labels for 4,149 objects, which was created by professional scientists working on the Snapshot Serengeti project. To evaluate the proposed approach it took all labels received from the crowd for the 4,149 objects in the gold standard. They amount to 112,027 labels submitted by 8,304 volunteers. The second dataset is from the DarkSkies app within the Cities at Night project. It consists of 1,275,354 classifications by 19,818 volunteers submitted in a time span from April 27th, 2014 until December 5th, 2016. The gold standard consisted of 200 objects whose labels were manually validated by the science team in Cities at Night. These 200 objects received 1,341 labels from 692 labellers. The workflows for the two datasets are depicted in Figure 3.4 and 3.5 respectively.

4.3.2 Experiment 2: Extended Investigation on Different Workflows Under Volunteer and Paid Contexts

4.3.2.1 Data

For the second part of the experiment, it replicates two existing workflows from citizen science projects and deploys the tasks in Figure Eight platform, the answer inference architecture proposed in this study was then applied on datasets resulted from the paid crowdsourcing classification. The two workflows this study used are depicted in Figure 3.4 and 3.6 respectively. The first workflow is from the GalaxyZoo project in which to complete a classification task, it takes from a minimum of one step to a maximum of nine steps depending on options chosen at each step. The second dataset is from the DarkSkies app within the Cities at Night project and requires a minimum of one step and a maximum of three steps to classify an image. The gold standard consisted of 500 objects from GalaxyZoo and 200 objects from DarkSkies whose labels were manually validated by the authors and the science team in Cities at Night. The experiments are executed on Figure Eight with these two existing workflows to obtain datasets under two settings which are elaborated in section 4.3.2.2.
4.3.2.2 Experiment Setting

The aforementioned two selected workflows are implemented in this work without adding additional steps to classify each object (image in both cases). However, previous studies also constantly show that using test questions to train and filter the crowd workers before allowing them to continue with real tasks is efficient in improving quality and has been used as a widely accepted general design principle (Kittur et al., 2008; Heer and Bostock, 2010; Thaler et al., 2012; Zhuang et al., 2015; Zheng et al., 2017). Therefore, it is necessary to implement the selected workflows in two different settings, with test question versus without test question, so that it may present a better picture on whether it also affects the performance of aggregation strategy and corresponding algorithms. As a result, 4 jobs in total are implemented in Figure Eight platform: two jobs for GalaxyZoo workflow, in which one has test questions; another two jobs for Darkskies workflow with one of them having test questions. It is worth mentioning that the test mode has the same user interface as the real task in Figure Eight, it is simply a pre-screening/training process to recruit qualified crowd workers. The implemented jobs’ UI in Figure Eight for GalaxyZoo workflow and DarkSkies workflow are shown in Figures 4.3 and 4.4 respectively. Following are the detail job settings:

**Judgments:** This experiment sets 7 judgments for Darkskies and 15 judgments for GalaxyZoo by default for each image to be classified, which is based on the corresponding settings each project has on citizen science platform.

**Contributors:** As there are two jobs with same images to classify for each workflow, it makes sure contributors for the two jobs are from different channels. For both workflows, the jobs without test questions use crowd workers from the same channels. The other two jobs with test questions are allocated to use the workers from the remaining channels.

**Quality control:** Test questions are created for two jobs (25 for GalaxyZoo and 10 for Darkskies) and require contributors to pass a specified minimum accuracy rate 60% before working on the units. In light of objects used in the test questions, they are different than the objects (500 objects from GalaxyZoo and 200 objects from DarkSkies) for the classification tasks and gold standards are obtained from the authors and peers in the same research group.

**Task:** It has one image per task by default which is consistent with the original GalaxyZoo and Darkskies workflow. However, due to the restriction on Crowflower where test mode and task mode have the same settings, including the same number of units (image) per page (task), it sets 3 images per task for the jobs with test questions, in order to select qualified crowd workers beforehand.

---

Payment: For each task consisting of 1 unit, it paid 1 cent for Darkskies task and 4 cents for GalaxyZoo task. This setting is chosen by taking into account existing surveys, as well as related experiments which have similar complexity level Kittur et al. (2008); Ipeirotis (2010); Acosta et al. (2013b). The DarkSkies task is relatively simple and easy to complete as it contains less steps and the only path involving a maximum of 3 steps is when the image contains a ‘city’. On the other hand, GalaxyZoo task is a bit difficult involving up to 9 maximum steps when the object has a certain feature in it and hence needs more effort to complete the classification.

4.3.3 Experiment 3: Further Investigation of Multiple-step Classification Aggregations

To comprehensively evaluate the three algorithms and the adapted approach this study has explored, it compares the classic approach where algorithms are applied on each node level and simply put together (referred as “naive-approach” here) with the “adapted-approach” which utilises classic approach while strives to infer a valid correct path by considering the workflow graph – on all three workflows seen in both 4.3.1 and 4.3.2. Thus there are six different approaches: \textit{mv\_adapted, mv\_naive, mp\_adapted,}
mp naive, em adapted, em naive. Each inference algorithm was applied to six datasets with different microtask crowdsourcing workflows. To do this, we also need to replicate Snapshot Serengeti\textsuperscript{15} workflow (as it is done in Experiment 2 for in section 4.3.2 for Cities at Night\textsuperscript{36} and GalaxyZoo\textsuperscript{14}). UI for the replicated Serengeti workflow is shown in Figure 4.5.

### 4.3.4 Results

#### 4.3.4.1 Experiment 1 Results

<table>
<thead>
<tr>
<th></th>
<th>MV</th>
<th>MP</th>
<th>EM</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.614, 0.785</td>
<td>0.004, 0.6625</td>
<td>0.444, 0.5875</td>
<td>run-on-path</td>
</tr>
<tr>
<td></td>
<td>0.562, 0.4925</td>
<td>0.526, 0.45375</td>
<td>dependency-aware adapted</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MV</th>
<th>MP</th>
<th>EM</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.776, 0.945</td>
<td>0.017, 0.665</td>
<td>0.645, 0.615</td>
<td>run-on-path</td>
</tr>
<tr>
<td></td>
<td>0.756, 0.585</td>
<td>0.883, 0.905</td>
<td>dependency-aware adapted</td>
<td></td>
</tr>
</tbody>
</table>

The results for MARP are shown in Table 4.3. The simple majority voting (MV) algorithm outperformed the more sophisticated performance-based approaches in both datasets. Averaged DCG results tell a similar story, as shown in Figure 4.6.
Chapter 4 Aggregation in Multiple-step Classification

As most of the existing studies on inference algorithms also use accuracy to measure performance (Sheshadri and Lease, 2013; Quoc Viet Hung et al., 2013; Zheng et al., 2017), the accuracy is plotted in Table 4.4 based on the first three candidates in the ranked lists produced by each algorithm. It chose the top-3 candidate labels because of the workflows in the gold standards, which have either one or two labels - as such, the first three labels on the inferred ranked list are expected to include the true labels. At the same time, the precision does not change much past the top-3 results (Figure 4.7(a) and Figure 4.7(b)). Expectation maximisation achieves the highest accuracy, on the Snapshot Serengeti dataset. Message passing, on the other hand, performs relatively well in the dependency-aware adapted implementation but fails in the run-on-path implementation for the same dataset. Such a drastic performance difference could not be observed for the DarkSkies dataset. On the other hand, the dependency-aware variant of MP achieves higher MARP and a better accuracy on the Snapshot Serengeti dataset (with 2 levels of nodes) than on the DarkSkies dataset (with 3 levels of nodes). Additionally, the EM dependency-aware node-based adapted implementation reaches a higher accuracy compared with the run-on-path approach for both datasets.

As it can be seen in Table 4.4, when taking into account the top – 3 candidates determined via majority voting, the true label is not correctly inferred for many objects. When inspected the characteristics of these objects as shown in in Figure 4.8(a) and Figure 4.8(b), it is worth noting that the distribution of vote percentages for all objects

Figure 4.5: UI for the Serengeti task in Crowdflower
Figure 4.6: Averaged DCG (Discounted Cumulative Gain) for each algorithm

Figure 4.7: Precision at K

(upper sub-graph in each figure) is skewed and if the top – 3 candidates are considered, the vote fraction is almost approaching 1.0 for most objects. The lower sub-graphs in Figure 4.8(a) and Figure 4.8(b) show the distribution of vote fractions for those objects, which are not correctly inferred by the top – 3 list with MV. A further analysis on the objects where MV failed to infer the true label revealed that more than half (57.08% for the Snapshot Serengeti case and 50.00% for DarkSkies) can be inferred correctly by the other two algorithms, where the EM dependency-aware adapted implementation accounts for 87.60% and 83.33% of these objects respectively.
4.3.4.2 Experiment 2 Results

The results for MARP are shown in Table 4.5 and 4.6. It is obvious in both GalaxyZoo and DarkSkies workflow, it has a better result for the job with test questions compared with the job without any pre-conditions to participate in the classification. It can be also observed that the aggregation strategy of using run-on-path and dependency-aware adapted approach behaves differently for DarkSkies workflow and GalaxyZoo workflow: it seems run-on-path approach works better for DarkSkies workflow which has fewer steps (nodes in the path) while dependency-aware strategy outperforms run-on-path approach for the more complex workflow like GalaxyZoo. Figure 4.9 has the Averaged DCG results and shows MV and EM adapted consistently perform well in all cases.

Table 4.7 and 4.8 shows the accuracy based on the top-3 candidate labels, and tell a similar story that job with test questions have apparent higher accuracy than job without test question, which is true for both workflows. It is also worth noting that both MV and EM adapted have consistent higher accuracy over the other approaches in both workflows no matter with or without test questions.
4.3.4.3 Experiment 3 Results

Table 4.9 shows the accuracy of each algorithm on each dataset for the top-1 inferred answer. Considering the overall classification accuracy (by path), the adapted methods have better accuracy than the naive approach in both volunteer and paid crowd context; at the same time, each algorithm generally has higher accuracy for volunteer context.

**Table 4.7:** Accuracy of inferred label path (top-3 results) - DarkSkies (without/with test question)

<table>
<thead>
<tr>
<th></th>
<th>MV</th>
<th>MP</th>
<th>EM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.790, 0.910</td>
<td>0.400, 0.560</td>
<td>0.420, 0.575</td>
</tr>
<tr>
<td></td>
<td>0.350, 0.485</td>
<td>0.825, 0.900</td>
<td></td>
</tr>
<tr>
<td></td>
<td>run-on-path</td>
<td>adapted</td>
<td></td>
</tr>
</tbody>
</table>

**Table 4.8:** Accuracy of inferred label path (top-3 results) - GalaxyZoo (without/with test question)

<table>
<thead>
<tr>
<th></th>
<th>MV</th>
<th>MP</th>
<th>EM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.872, 0.871</td>
<td>0.002, 0.002</td>
<td>0.188, 0.516</td>
</tr>
<tr>
<td></td>
<td>0.369, 0.572</td>
<td>0.782, 0.857</td>
<td></td>
</tr>
<tr>
<td></td>
<td>run-on-path</td>
<td>adapted</td>
<td></td>
</tr>
</tbody>
</table>
compared to the paid crowd. Note that the best accuracy achieved increases as the depth of the workflow increases for the paid crowd context, where Serengeti with two questions achieves 45.9%, Darkskies with three questions achieves 53.0% and Galaxyzoo with a maximum of nine questions achieves 57.9%. A similar pattern is not observed for the volunteer context. If looking at the accuracy breakdown by node level (Figure 4.10, 4.11 and 4.12), it is notable that for multiple-questions task with more steps, dependency-aware adapted method of MP and EM generally shows better accuracy at most of the node levels. For the datasets from a task with fewer steps in its workflow (less number of levels in the graph), such as the Serengeti task in Figure 4.10, MV performs better.

<table>
<thead>
<tr>
<th>dataset</th>
<th>graph depth/size</th>
<th>crowd type</th>
<th>algorithm</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>serengeti</td>
<td>54-11</td>
<td>volunteer</td>
<td>mv_naive</td>
<td>0.590</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.776</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.572</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.655</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.755</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.755</td>
</tr>
<tr>
<td></td>
<td></td>
<td>paid</td>
<td>mv_naive</td>
<td>0.299</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.459</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.337</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.383</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.307</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.495</td>
</tr>
<tr>
<td>darkskies</td>
<td>8-3-2</td>
<td>volunteer</td>
<td>mv_naive</td>
<td>0.405</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.785</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.040</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.450</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.340</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.495</td>
</tr>
<tr>
<td></td>
<td></td>
<td>paid</td>
<td>mv_naive</td>
<td>0.530</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.385</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.020</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.335</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.305</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.305</td>
</tr>
<tr>
<td>galaxyzoo</td>
<td>3-3-2-3-2-2-3-6-4-2-7</td>
<td>volunteer</td>
<td>mv_naive</td>
<td>0.554</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.631</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.470</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.564</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.002</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.562</td>
</tr>
<tr>
<td></td>
<td></td>
<td>paid</td>
<td>mv_naive</td>
<td>0.371</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mv_adapted</td>
<td>0.579</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_naive</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em_adapted</td>
<td>0.331</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_naive</td>
<td>0.002</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp_adapted</td>
<td>0.367</td>
</tr>
</tbody>
</table>

Table 4.9: Accuracy (by path) of each algorithm

Similar to previous experiments, the average DCG graph is plotted (Figure 4.13) against all three datasets now both volunteer and paid crowd data are available, to give an overall view of whether the classification accuracy could be improved by taking into
### Figure 4.10: Accuracy by node level (Serengeti)

<table>
<thead>
<tr>
<th>Level</th>
<th>Accuracy</th>
<th>Paid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.977</td>
<td>0.979</td>
</tr>
<tr>
<td>2</td>
<td>0.924</td>
<td>0.919</td>
</tr>
<tr>
<td>3</td>
<td>0.834</td>
<td>0.819</td>
</tr>
<tr>
<td>4</td>
<td>0.740</td>
<td>0.760</td>
</tr>
<tr>
<td>5</td>
<td>0.549</td>
<td>0.549</td>
</tr>
</tbody>
</table>

*Note: Accuracy and Paid values are estimated.*
### Chapter 4: Aggregation in Multiple-Step Classification

#### Figure 4.11: Accuracy by node level (bars: colors)

<table>
<thead>
<tr>
<th>Node Level</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.70</td>
<td>669.036</td>
</tr>
<tr>
<td>0.75</td>
<td>669.036</td>
</tr>
<tr>
<td>0.80</td>
<td>669.036</td>
</tr>
<tr>
<td>0.85</td>
<td>669.036</td>
</tr>
<tr>
<td>0.90</td>
<td>669.036</td>
</tr>
<tr>
<td>0.95</td>
<td>669.036</td>
</tr>
<tr>
<td>1.00</td>
<td>669.036</td>
</tr>
</tbody>
</table>

#### Diagram:
- Nodes represent different node levels.
- Edges indicate dependencies or relationships.
- The table above correlates accuracy with node levels.
Figure 4.12: Accuracy by node level (Galaxyzoo)
consideration of more top inferred answers. Noted the data chosen here has the same condition: volunteer case and the paid crowd case (without using test question, as the experiment 3 with random selected Serengeti objects are crowdsourced without test question). As seen from the graph, if taken the first few top inferred results (for example top-3), it seems could produce better accuracy. It is also notable that in most cases, MV (shown in green line) consistently performs well.

Meanwhile, table 4.9 shows MV has an acceptable accuracy for most of the volunteered datasets (mostly over 75%, except for GalaxyZoo dataset), but has poor accuracy (less than 60%) in the paid crowd context though it performs better than other individual algorithms tested here, which suggests it need to be complemented by other methods which might be good at specific objects where MV cannot perform well. Looking at the accuracy by level results, it does not seem to suggest that as the depth of the task (number of levels) increases, accuracy has a tendency to consistently increase or decrease. The accuracy of each level is more relevant to its intrinsic character (eg, number of options in that level, and ambiguity or subjectivity of the corresponding object). For instance, the Darkskies task asks the user to evaluate the sharpness and cloudiness of the image, which can be subjective to some degree. This is also why the result by node level seems to show an interesting picture that on different node level for different workflow, sometimes em has the best result (such as level 4 and 5 of GalaxyZoo), sometimes mp
has the best result (such as level 1 of Serengeti in volunteer case), other times \textit{mv} has the best result (level 1,2,3 of Darkskies in both volunteer and paid context).

Notice that MP for the Darkskies paid crowd context, it is the only case where the \textit{naive} approach has higher overall accuracy (by path) than \textit{adapted}, which is due to the fact that both the level 2 and 3 (determining cloudiness and sharpness of the image) of Darkskies workflow are in essence independent questions of the first node level (whether it is a city, or stars or anything else) though the task workflow made it a subsequent question only when “city” is chosen as the label for first node level. Similarly, the accuracy by level result from \textit{mp\_adapted} is lower than \textit{mp\_naive} on a few other occasions at different node level, but in those occasions, there is always one node level \textit{mp\_naive} has considerably poor accuracy, such as in Galaxyzoo node level 2, which subsequently leads to the very low overall accuracy considering the whole path. The reason that the \textit{mp\_adapted} approach could have lower accuracy at certain level is that \textit{mp} approach actually only returns 1.0 or 0.0 to indicate whether that is the predicted label, but the adapted approach tried to assemble/infer a most probable valid label path (as shown in algorithm 4) based on the candidate of predicted labels from individual node level. So for the \textit{mp} case, the randomness of ranking the combinations might not do well for the corresponding node level, however, the overall accuracy has shown to be better than the \textit{naive} approach which completely neglects the validity of a label path.

Notice that though the dependency-aware \textit{adapted} approaches achieve higher accuracy for the first node level in most case, \textit{mv\_adapted} has slightly lower accuracy comparing to \textit{mv\_naive} for GalaxyZoo workflow under volunteer context, which is because the way it assembles the result is based on the overall possibility (percentage of voting at each node level multiplied) of a path instead of assuming the top voted label at node level 1 is correct (and then traversing subsequent node based on that assumption). The main purpose is to obtain the most possible valid label path, which has been shown effective in Table 4.9. Significant testing\textsuperscript{6} for all algorithms chosen is done (details in 4.2.4). The result is statistically significant for all the \textit{adapted} approach as the p-value is smaller than the pre-defined significant level (5\%) in all cases.

The experiments show that majority voting, despite its simplicity, compares well with more sophisticated approaches when the crowd has been “trained” and has a better understanding of the task, but sophisticated algorithm such as expectation maximisation outperforms the others for every workflow when the crowd is with uncertain level of expertise in the domain area. The results also seem to suggest that the structure of the microtask workflow, in terms of the number of answer options and the number of questions per workflow impacts the performance of each algorithm.
4.4 Discussion

Looking into the existing classification examples, it seems in general there are two main considerations that need to be taken into account when designing a microtask crowdsourcing workflow: the number of possible paths, which is heavily influenced by how many answer options each question allows; and the length of these paths, which is related to the number of questions in the workflow. In the evaluation, it has found evidence that both impact on the overall performance of the inference architecture.

A critical finding is that one aggregation method does not perform consistently on different datasets resulting from different workflows. The example workflows this study has chosen are representative: one is relatively simple with maximum 3 steps and minimum 1 step, and maximum 7 options in one level; one is fixed 2 steps but one of the steps involves more than 50 options; the other one is comparatively complex which has more dependent questions and more steps. This may mean that quality assessment in multiple-step classifications is not agnostic of workflow properties, such as the number of steps in the workflow, the number of options to choose from for each question, or, as discussed elsewhere (Sprinks, 2015; Simperl et al., 2011b; Yang et al., 2016; Sigurdsson et al., 2016) the difficulty of individual microtasks. The observation so far indicates run-on-path approach performs better on simple workflow (Table 4.5) with the $MP$ algorithm. Comparing to run-on-path approach, dependency-aware adapted aggregation works better on multiple-step classification (Table 4.6 and 4.8) for both $MP$ and $EM$ algorithms. For the Darkskies dataset, the result from the volunteered crowd and the paid crowd who have been selected by passing test question shows MARP (0.785,0.53,0.694) and Accuracy (0.945,0.79,0.91) for $MV$ which indicate $MV$ can be used when the crowd has been trained or has less chance of being spammers. $EM$ dependency-aware strategy on the DarkSkies workflow has MARP (0.454,0.384,0.452) and Accuracy (0.905,0.825,0.900)) which shows a comparable performance of $MV$ and clearly have higher accuracy when data might be noisy (0.825 vs. 0.79 in paid crowd setting).

Along those lines, we can see that applying established inference models on entire label paths (the run-on-path approach) leads to a very low accuracy in GalaxyZoo and Serengeti compared to the more nuanced dependency-aware adapted condition that rewards partial correctness. The difference between GalaxyZoo/Serengeti and DarkSkies datasets lies especially in the number of possible paths in the workflows. The run-on-path $MP$ implementation needs to iterate over every possible path, in which for each possible path, the worker message might not contain enough information to infer the object label with so many possible paths and far fewer objects classified by each crowd worker. This might indicate the performance of $MP$ algorithm, no matter run-on-path or dependency-aware aggregation, is heavily affected by the number of labellers, the
number of objects each labeller classified, as well as the total number of available options, thus not suitable for the context where crowdsourcing tasks are done by a large number of contributors instead of a small fixed group of contributors. At the same time, the experiment shows the accuracy of $MV$ and $EM$ dependency-aware adapted are around 80% or higher in both simple one-step classification and multiple-step classification and could also be leveraged as a part of another classification workflow, such as using the top 3 results produced by these algorithms for the predictor stage and followed by error detection/correction stages as used in the framework proposed in Chapter 5 to achieve better classification result.

In general, the evaluation seems to suggest that considering the workflow structure (how the path is shaped level by level) is a more appropriate way to identify good performance for message passing and expectation maximisation algorithms, compared to naively applying algorithms on the entire path. The ranked lists of candidate paths produced by $MP$ and $EM$ in the dependency-aware adapted implementations can be computed more efficiently than their run-on-path counterparts for multiple-step classifications. This should be taken into consideration when designing a combined aggregator (strategy as demonstrated in section 4.2.3) and it might be a good idea to merge results from existing algorithms implemented in adapted instead of run-on-path mode. In a simple one-step classification task, run-on-path and dependency-aware adapted implementation is essentially the same. In a multiple-steps task, run-on-path and adapted implementation can exhibit different merits. The run-on-path approach basically does not need adaptation for the individual algorithm and each classification path is treated as a single entity, therefore it is intuitive to use and benefits from one-off execution. In contrast, adapted implementation needs to run multiple times on each level based on the workflow graph of the corresponding task and needs the additional assembling step to produce the ranked list of (valid) paths based on the graph. Though both $EM$ and $MP$ have the advantage of being able to take into account the user performance, in most of the classification tasks, not all workers contribute on all objects, this leads to two limitations: on one hand, many workers may have only worked on a few objects (regardless of a large number of objects needed to be classified) and hence the worker response is sparse; on the other hand, in run-on-path approach, it is less possible users agree on same label path comparing with that users agree on the same node and as a result, the object response is even more sparse in this case. With such limitation in mind, the straightforward run-on-path approach is likely to perform poorly compared with the dependency-aware adapted strategy utilising relatively “dense” labels for individual nodes.

However, this is by no means a one-size-fits-all solution. For the Darkskies dataset produced by the crowd with mixed expertise (without being selected by passing test questions), the node-based dependency-aware $EM$ implementation works best, followed by majority voting. It indicates, to some degree, a sophisticated algorithm such as
EM used in a dependency-aware adapted implementation seems to be a better choice in the case where the crowd’s knowledge level is uncertain. On the other hand, for the GalaxyZoo dataset, where path length can be much longer, MV seems unbeaten, with dependency-aware adapted EM as a runner-up. Although MV shows consistent good performance in all four datasets, the dependency-aware algorithms can be a supplement to MV when MV fails to infer the correct label. When inspected the objects that are not inferred by MV, it found that 21.6% for the GalaxyZoo case and 37.2% for DarkSkies can be inferred correctly by the EM adapted implementation. This might also indicate some characteristics of corresponding objects that make it impossible to reliably infer the correct label which should be further considered apart from the workflow structure proposed in this paper. More research is needed to understand not only the effect of path length - the number of questions in a workflow sequence - on the ability to infer correct answers using suitable algorithm and aggregation strategy, but also the characteristics of objects that may require different automatic or hybrid approach to identify such objects and estimate their true labels.

Furthermore, on the node-based method, apart from the dependency-aware approach this research proposed, there is also a naive way of assembling results from each node level. Experiment 3 focuses on finding how naive assembling strategy performs and does show the dependency-aware adapted performs significantly better than naive. On observing the result in section 4.3.4.3, it seems to be a promising way if we consider combining output from these algorithms using a heuristic strategy to perform better inference. It could use results from mv adapted, em adapted and mp adapted in combination in order to exploit their strengths and weaknesses for complex classification tasks. To do so, an aggregator could be considered which is based on following intuitions: 1. the number of unique classifications of an object (defined by u) shows the degree that the crowd workers agree/disagree on the classification where the higher number indicates higher degree of disagreement and normally imply the object is either a bit difficult or ambiguous to be classified. 2. the ratio (defined by r) between the unique number of classifications/answers collected from the crowd and the total number of classifications/judgments also demonstrates how diverse the answers are for the corresponding object and hence similarly. 3. As three-sigma rule Pukelsheim (1994) in the empirical sciences suggests that almost all values should lie within three standard deviations of the mean in a normal distribution, and theoretically mean plus one, two or three standard deviation(s) covers 68%, 95% and 99.7% of the data. In the case where majority voting might potentially fail (where workers tend to disagree), the number of unique classification or the ratio of the number of unique to the total number of classification for an object falls within the higher range of the distribution. Thus, a heuristic aggregation strategy is worth considering: Look at the intrinsic characteristics of collected classifications for each object, such as the number of unique classifications and the ratio of that against the total number of classifications. Then, based on the third intuition above, it can utilise the skewness (defined by s below) of the distribution for number of unique
$(U \sim N(u_\mu, u_\sigma))$ and ratio (defined by $R \sim N(r_\mu, r_\sigma)$) respectively to heuristically chosen bound where $MV$ can be potentially complemented by other approaches. However, choosing an optimal threshold is not straightforward and needs to be explored in future work.
Chapter 5

Classification Workflow

This chapter elaborates the classification model stated in section 3.6 and takes the first step to instantiate the model with a user case to explore alternative workflows. The model is tested with an entity classification scenario, in which specific type of workflow is chosen based on the specific type of objects (here is “DBpedia Entities”) accordingly. The performance of alternative workflows are evaluated, based on the predictor, error detector, error corrector and the average number of the steps it takes.

5.1 Overview

DBpedia is at the core of the Linked Open Data Cloud and widely used in research and applications. However, it is far from being perfect. Its content suffers from many flaws, as a result of factual errors inherited from Wikipedia or incomplete mappings from Wikipedia infobox to DBpedia ontology\(^1\). This work focuses on one class of such problems, un-typed entities. The main contributions of this work are as follows: It proposes a hierarchical tree-based approach to categorize DBpedia entities according to the DBpedia ontology using human computation and paid microtasks. It analyses the main dimensions of the crowdsourcing exercise in depth in order to come up with

\(^{1}\)http://mappings.dbpedia.org/index.php/Use_the_DBpedia_Extraction_Framework

97
suggestions for workflow design using the classification workflow model and studies three different workflows with automatic, hybrid and human prediction mechanisms to select possible candidates for the most specific category from the DBpedia ontology. To test the approach, experiments are run on Figure Eight\footnote{In this context, cost refers to the efforts (number of steps) it takes to complete a task.} using a gold standard dataset of 120 previously unclassified entities. This work shows that human-computation driven approaches generally achieved higher precision with lower effort cost\footnote{In this context, cost refers to the efforts (number of steps) it takes to complete a task.} when compared to workflows with automatic predictors.

### 5.2 Classification Workflow Model in Entity Classification

![Diagram of three workflows with human participation steps highlighted in blue.](image)

The entity classification problem considered in DBpedia entity classification case is a problem of selecting the most specific type from a given class hierarchy for a particular entity. The problem can be formalised as follows: Let $W_O$ be the given DBpedia ontology and $o$ be a particular entity which has not been classified in DBpedia. The aim is to find the most specific type $N_E$ for entity $o$ within $W_O$. For an un-typed entity, it considers this as a tree traversal problem starting from a predicted candidate node in the ontology and continued until the most specific type is found. To this end, as depicted in Figure 5.1, this work break the workflow into (a) prediction step, where a list of candidate classes is identified first, (b) error detection step where the output is manually checked and (c) error correction step where the error (if detected) is manually corrected. This study varies the implementation of each step and measures the performance of the proposed model by two of the most important factors that play into the success of a
microtask approach to DBpedia entity typing: precision and costs of efforts (measured by the number of steps taken). Precision refers to the ability of crowd contributors to submit precise answers and the objective is to minimise the costs denoted by the amount of manual work required in a workflow. To be more specific, the cost in this model is the sum of prediction, error detection, and error correction effort, measured in terms of the number of steps to finish the corresponding tasks by crowd workers. Monetary cost or time cost is not specially considered in the evaluation, as the assumption here is the efforts it takes to complete a task correlates with the difficulty of the task, which accordingly affects the monetary and time cost.

5.2.1 Formalization

This section formally defines the classification model which is a human computation driven model for entity annotation using the hierarchically structured set of labels. It starts with the definition of the terms used in the classification model under entity classification context, then covers the main tasks where human participation is involved – predictor, the subsequent error detection and error correction process, and how the quality and cost of efforts are evaluated.

Definition 5.1 (DBpedia Ontology). The DBpedia ontology $W_O$ is a tree structure, where each node corresponds to an entity category and can contain a list of references to the child nodes, each of those corresponds to a more specific sub-category of the parent entity type. The root node $ROOT$ (“Thing”) is the topmost node in the ontology.

Definition 5.2 (Candidate Nodes). Given an entity object $o$ and ontology $W_O$, particular node $N$ is considered as the exact solution node $N_E$ iff $N$ and not any of the children nodes of $N$ is corresponding to the category of $o$, i.e $N$ represents the most specific category for $o$ in $W_O$. Any ancestor of $N_E$ thereby is considered as a candidate node $N_C$.

Definition 5.3 (Predictor and Predicted Nodes). Let a predictor $P$ be a manual or (semi-) automatic process, able to select for given $o$ a ranked set $S_{predicted} = \{N_{p1}, \ldots, N_{pn}\}$ of nodes from $W_O$ as predicted candidates for $N_E$.

The assumption is that the location of an $N_P$ within $W_O$ is close to $N_E$ with high probability. In case no prediction can be made by $P$ for a particular $o$, it considers $S_{predicted} = \{ROOT\}$, as the $ROOT$ is the closest to the optimal solution given the hierarchical structure (see definition in 5.1).

**Human computation driven error detection and correction model:** The error detection and correction process in this context are to traverse $W_O$ starting from a set $S$ consisting of top-ranked predicted nodes $N_P$, until $N_E$ is found and to break the process if no correct solution can be found. Each traversal step can be viewed as a
classification microtask for a human assessor, where it asks, whether $N_C$ exists in a list of options. The answer can be true (with an indication of the corresponding node) or false, in case no nodes from the list can be selected. Generally, it assumes that if a node is proven to be false, also all of its descendants are proven to be false.

- **Error detection algorithm**: A traversal mechanism is employed as indicated in Figure 6 with a set $S$ of the top-scored nodes as a start. In case any of $N_P$ from this set can be identified as $N_C$, a check is done for each of its child nodes according to the definition 5.2.

- **Error correction algorithm**: After the error is detected, it is possible to correct it using human assessors. Similarly to error detection, the microtask based correction (algorithm 7) starts from the set $S$ of candidate nodes. In case a $N_P$ from $S$ is identified as $N_C$, its child nodes are traversed in a breadth-first manner until the node with the most specific type corresponding to $o$ is found. Otherwise, the algorithm continues with the parent node of $N_P$. Every node on the way is touched only once. The algorithm stops when $N_C$ is found without children corresponding to the type of $o$. Note, in case no specific node can be found in $W_O$, the algorithm will return $ROOT$.

**Algorithm 6** Error detection

1: procedure CHECKSPECIFICTYPE($o$, $S = \{N_{p1} \ldots N_{pn}\}$)
2: if $N_c \in S$ and humanChoice() = $N_c$ then
3: \hspace{1cm} if $\forall N_i \in$ children($N_c$), humanChoice() $\neq N_i$ then
4: \hspace{2cm} return TRUE
5: return FAIL;

**Cost of efforts**: The overall effort cost for entity classification in this model is constituted as the sum of prediction, error detection, and error correction costs, more formally:

\[
Cost(classification) = Cost(prediction) + Cost(detection) + Cost(correction)
\]

In which, the costs of a particular algorithm are defined as the number of steps necessary to complete the algorithm run. Cost(detection) and Cost(correction) will be the cost to detect and correct an error, which corresponds to the steps it takes to run the above Algorithm 6 and 7 respectively. The details of the prediction step is described in the next section.
Algorithm 7 Error correction

1: procedure FINDSPECIFICTYPE(o, S = \{N_{p1} \cdots N_{pn}\})

2: if \(N_c \in S\) and humanChoice() = \(N_c\) then

3: \(C = \text{anyChild}(o, N_c)\)

4: if \(C = \text{FAIL}\) then

5: return \(N_c\)

6: else

7: return \(C\)

8: else

9: \(S_{\text{parents}} \leftarrow \{\}\)

10: for \(N_i \in S\) do

11: \(S_{\text{parents}} \leftarrow S \cup \text{parent}(N_i)\)

12: return findSpecificType(o, \(S_{\text{parents}}\))

13: procedure ANYCHILD(o, N)

14: for \(N_i \in \text{children}(N)\) do

15: if \(N_i = \text{humanChoice()}\) then

16: \(N_{\text{child}} = \text{anyChild}(e, N_i)\)

17: if \(N_{\text{child}} = \text{FAIL}\) then

18: return \(N_i\)

19: else

20: return \(N_{\text{child}}\)

21: return FAIL;

5.2.2 Predictors

A predictor, in this case, is a module, able to produce a list of candidate classes for a given entity. Currently, all the predictors described in the literature are automatic predictors, where given an entity a list of candidates along with the confidence value of the predictions is produced. This work employs three approaches, namely an automatic predictor \(P_{\text{auto}}\), and two human-based predictors \(P_{\text{naive}}\) and \(P_{\text{hybrid}}\) as follows:

As an example for \(P_{\text{auto}}\), it uses DandelionAPI\(^3\) which has an add-on for Google Spreadsheet allowing entities uploaded in a spreadsheet to be analysed and typed/classified with DBpedia types. The entity to be classified is the text to be analysed, and the output is the types along with the confidence levels. The types given from Dandelion API always include the parent types if a specific type is identified. For instance, if an entity’s specific category is “Building”, Dandelion would output Building, ArchitecturalStructure, and Place. In this case, it uses the open source tool which is free and the prediction cost can be considered as 0.

Additionally, two human computation based prediction approaches are proposed. The first \(P_{\text{naive}}\) is a completely human based prediction process, namely a “naive predictor” approach that starts from the root of the ontology and traverses the tree by sequentially

\(^3\)https://dandelion.eu/docs/api/
expanding the children based on the crowd’s choice. This approach can be applied also for entities, where any of the other (semi-) automatic approaches fail. The second human computation based prediction approach $P_{\text{hybrid}}$ allows for unconstrained input from the crowd, predicting the candidates from the collected freetexts with the help of automatic tools. The main advantages of this approach are its simplicity and freedom of choice. Moreover, restrictions on category input can sometimes increase the difficulty of task (Vickrey et al., 2008) and hence impact the overall accuracy. As a result, it might be a balanced approach that could reduce the efforts for the crowd as in naive predictor, but providing better result than automatic predictor.

For human-based predictor, the outcome depends on how many times the questions are asked and how the answers are aggregated. The more answers are collected, the more reliable is also the classification (Lintott et al., 2008). Since the focus of this work is the workflow model, straightforward majority aggregation algorithm is applied for quality assessment considering the good classification accuracy it can achieve (as seen in Chapter 4). Top aggregated answers can be voted by the crowd in a second step to identify the most suitable candidate (Little et al., 2010b), or detect errors and correct the answers, as it is demonstrated in this study. Considering the diversity of vocabulary of the crowd users in $P_{\text{hybrid}}$, direct aggregation of their answers is not effective. To solve this issue, this study leverages the freetext input to automatically calculate the closest DBpedia types based on the textual similarity between entity titles and ontology class names. It uses the difflib\(^4\) SequenceMatcher to compute a string match similarity scores between every input collected from the crowd and names of all of existing DBpedia classes. Each DBpedia class is assigned an aggregated score as the indication of how close it is to the user proposed type. Finally the top scored DBpedia types are retrieved to form the list of output candidates to be used in error detection and error correction steps.

### 5.2.3 Workflows

In the classification workflow model (Figure 3.10), this study defines three different workflow as $W_{\text{auto}}$, $W_{\text{naive}}$, $W_{\text{hybrid}}$ based on the type of predictor each employs.

- **($W_{\text{naive}}$) Naive**: ask the crowd to choose the DBpedia category by traversing from the root top-down until a specific category is selected. This is the workflow that employs $P_{\text{naive}}$ predictor.

- **($W_{\text{hybrid}}$) Hybrid**: accepts unconstrained input from the crowd to label an entity and combined with the use of automated tool to come up the predicted candidate list. This workflow incorporates the $P_{\text{hybrid}}$ predictor where collected freetexts annotations are processed to identify the top candidates in DBpedia types.

\(^4\)https://docs.python.org/2/library/difflib.html
which then can be explored and corrected by the crowd in a follow-up verification task.

- **(W_{auto}) Automatic**: uses an entity typing tool to generate candidate list, then ask the crowd to detect and correct errors. This workflow is based on the \( P_{auto} \) predictor.

Figure 5.1 not only shows an overview of these three workflows but also highlights the steps requiring human participation in blue background. The remainder of this section explains the workflows and their translation into microtasks in more detail.

For the **(W_{naive}) Naive** workflow, the idea is to traverse the DBpedia class hierarchy top-down. The particular worker choice from a list depends on her level of expertise and on the given situation, as the theory teaches. Rosch et al. proved in experiments that experts and newbies make very different classification decisions - people with little insight into a domain tend to feel comfortable with categories that are neither too abstract, not too specific, whereas experts are much more nuanced (Rosch et al. (1976)). The same effect was observed by Tanaka and Taylor (1991) or in games with a purpose (Von Ahn and Dabbish (2004)). As this task is executed on the microtask platform\(^2\), it has to assume that the behaviour of the crowd contributors will resemble newbies in the Rosch et al. experiments and casual gamers who interacted with GWAPs. Thus, we cannot always expect from \( P_{naive} \) to identify the most specific class in the DBpedia ontology, which matches the input entity. This is also why in each list of options presented to the crowd, it needs to include a NoneOfAbove option. In this Naive workflow, the crowd is firstly presented with a list of candidate types (first level of DBpedia classes under the ROOT, along with NoneOfAbove). As there are 49 first level classes in DBpedia, using the naive approach, the classes at each level are randomly put into groups and one such group is presented at a time. The details of the decision on the number of options to be put in each group are available in the task design section 5.3.1. If NoneOfAbove is chosen, another group of options is presented to the worker. If any given level 1 class is chosen and the class has children, a group candidate types from level 2 class will be displayed for the worker instead. The process keeps going until a class which has no children is chosen, or a chosen class does not contain children that are suitable specific types for the given entity. The result of this process is considered as final.

The **(W_{hybrid}) Freetext-based Hybrid** workflow starts with a task where an entity and its description are presented to the worker and free text annotations from this worker are collected. Once annotations for all entities are collected, for each entity, it considers all the annotations and calculate their textual similarity (many available APIs\(^5,6,7\)) with titles of all existing DBpedia classes(vocabularies) and add-up a similarity score for each

---

\(^{5}\) https://pypi.python.org/pypi/Distance/  
\(^{6}\) https://docs.python.org/2.7/library/difflib.html  
\(^{7}\) http://www.nltk.org/howto/wordnet.html
Chapter 5 Classification Workflow

DBpedia classes. The higher the similarity score of a DBpedia class has, the higher is the chance that this class is the candidate type of the corresponding entity. Then the top classes for each entity are picked and add a “NoneOfAbove” option to form a shortlist as the predicted result from \( P_{\text{hybrid}} \). Then it starts the error detection and correction process. Similar to \( W_{\text{naive}} \), it traverses the DBpedia class hierarchy until a class which has no children is chosen, or a class has been chosen, but none of its children are suitable specific types for the given entity. In the case when none of the predictor predicted classes is chosen as a suitable type for the given entity, we will traverse up to the parent level. Depending on whether a suitable class is chosen from the parent level or not, further child or parent classes will be presented respectively. The process keeps going until it reaches ROOT or a class is chosen and that class either has no children or none of its children are suitable types for the given entity.

For the \( W_{\text{auto}} \) Automatic workflow, the idea is to use one of existing APIs\(^8\),\(^9\),\(^10\) to produce a list of candidate types for the given entities, and only the top types with higher confidence level are picked to be presented along with the NoneOfAbove option to the crowd. Based on worker’s selection, the similar traverse process is followed until it reaches ROOT or a class is chosen and that class either has no children or none of its children classes is chosen.

5.3 Applying the Model to DBpedia Entity Classification

5.3.1 Microtask Design

In general, this study distinguishes between two types of microtasks based on their output: \( T_1 \) where the workers produce free text output; and \( T_2 \) where the worker can choose from a list of classes. In both cases, it generates descriptions of the input entities in the form of labels or text summaries. Either way, the effort to generate the first type of tasks (\( T_1 \)) is comparatively lower, as there is no need to compute a list of potential candidates. However, this is compensated by overhead in sense-making of the output and means to aggregate free text inputs into meaningful class suggestions, while in \( T_2 \) the answer domain is well-defined. Crowdsourcing literature recommends to use iterative tasks to deal with \( T_1 \) scenarios: in a first step, the crowd is generating suggestions, while in the second it is asked to vote on the most promising ones (Brown and Allison (2014b); Chilton et al. (2013); Little et al. (2010b)). Accordingly, \( T_2 \) can be used as the voting step for \( T_1 \) output to get more useful suggestions in the condition \( T_1 \) output has been post-processed to produce a ranked list, such as using the aggregation we proposed in 5.2.2 for \( P_{\text{hybrid}} \).

---

\(^8\) https://dandelion.eu/docs/api/
\(^9\) http://www.alchemyapi.com/api/entity/types
\(^10\) http://www.opencalais.com/opencalais-api/
The **T2** variant requires the strategy to generate the list of candidates. It can be achieved through automatic tools addressing a similar task (as listed in section 2.2.2), however, they impose clear bounds on the accuracy of the crowd experiments, as not all input can be processed by the automatic tools (recall) and their output can only be as precise as the task input allows (precision). Additionally, the choice of an appropriate threshold to accept or reject produced results is also a subject of debate in related literature (Scheibehenne et al., 2009; Schwartz, 2004). Another option is to provide the crowd all possible choices in a finite domain - in our case all classes of the DBpedia ontology. The challenge then is to find a meaningful way for the crowd to explore these choices. While classification is indeed one of our most common human skills, cognitive psychology established that people have trouble when too many choices are possible. This means both, too many classes to consider (Iyengar and Lepper (2000); Schwartz (2004)) and too many relevant criteria to decide whether an item belongs to a class or not (Alfonso-Reese et al. (2002b); Hua et al. (2005)). Miller’s research suggests that in terms of the capacity limit for people to process information, 7 (plus or minus 2) options are a suitable benchmark (Miller (1956)). This study hence would ideally use between 5 and 10 classes to choose from. This situation is given by the predictor \( P_{\text{auto}} \), as the confidence of automatic entity typing algorithms decreases rapidly and only the top 10 are likely to be representative. However, in the \( P_{\text{naive}} \) condition, we start from the DBPedia ontology, which has over 700 classes to choose from\(^\text{11}\). The problem persists even when browsing the ontology level by level, as some classes have tens of subclasses (e.g., there are 21 different forms of organisation, 50 child categories of person and 43 specific types of athlete). Therefore, in all the experiments of this study, it splits the list into subsets of 7 items or less to display per microtask.

### 5.3.2 Implementation

In the classification model, all three workflows take the same input. As a result, for each workflow, it first queries the DBpedia endpoint via SPARQL to obtain the name, description, and a link to Wikipedia of each entity. \( W_{\text{naive}} \) and \( W_{\text{auto}} \) workflows only involve T2 task type, \( W_{\text{hybrid}} \) include both T1 and T2 in sequence. The Figure 5.2 and 5.3 depict the user interfaces for T1 and T2 types of tasks. For T2 it has always limited the number of options shown to a user in the list to a maximum of 7 (6 class candidates and a NoneOfAbove option). In case there are more candidates, it splits the list as discussed earlier. The gold standard data (see Section 5.4) is created beforehand. The jobs are deployed on Figure Eight\(^2\) with the following settings.

**Task:** Following the advice from the literature (Acosta et al. (2013b)), 5 units/rows for each task/page for each of the three workflows is used. The worker completes a task by categorising five entities.

---

Judgment: Snow et al. (2008) claim that answers from an average of four non-experts could achieve a level of accuracy parallel to NLP experts on Mechanical Turk. It hence asks for 5 judgments per experimental data throughout our experiments. In $W_{hybrid}$, it asks for 11 free text suggestions. This choice is in line with experiments from the literature (Acosta et al. (2013b); Sarasua et al. (2012)). An empirical simulation with previous experiments on entities that have been classified is performed to find out the effective number of judgment to use. Figure 5.4 shows the number of times the entity has been annotated and the corresponding matched answers (based on 120 entities).

Payment: 6 cents are paid for each task consisting of 5 units. This setting took into account existing surveys (Ipeirotis (2010)), as well as related experiments which have a similar complexity level. Tasks like reading an article and then asking the crowd to rate the article based on given criteria as well as providing a suggestion for the areas to be improved were paid 5 cents (Kittur et al. (2008)). Tasks that have smaller granularity such as validating the given linked page display relevant image to the subject were paid 4 cents per 5 units (Acosta et al. (2013b)). In a similar vein, the complexity of
our classification task is somewhere in between considering the time and knowledge it requires to complete the task.

**Quality control**: Test questions are used for all jobs and contributors are required to pass a minimum accuracy rate (this study sets 50%) before working on the units.

**Contributors**: On Figure Eight platform, it distinguishes between three levels of contributors based on their previous performance. The higher level of contributors required, the longer it takes to finish the task, but might be of higher quality. In these experiments, default Level1 is chosen which allows all levels of contributors to participate in the classification task. This same level is used for all three workflows.

**Aggregation**: For the T2 tasks it uses the default option (aggregation=‘agg’)

5.4 Evaluation

This section evaluates the performance of proposed workflows as depicted in Figure 5.1. Overall, it obtained three different workflows, namely: \( W_{\text{naive}} \), \( W_{\text{auto}} \) and \( W_{\text{hybrid}} \), respectively based on predictors \( P_{\text{naive}} \), \( P_{\text{auto}} \) and \( P_{\text{hybrid}} \). It evaluates the accuracy of the predictors, and compares the overall precision and costs of different workflows.

---

Data For the experiments, it randomly and uniformly chooses 120 entities which were not classified so far by the DBPedia. These entities were annotated manually to obtain a gold standard. Two annotators worked independently and achieved an agreement of 0.72 measured using Cohen’s kappa\textsuperscript{13} (Cohen, 1960). According to one of the most commonly used interpretations by Landis and Koch(1977), kappa value in the range of 0.6-0.8 corresponds to a substantial agreement. Noted 106 out of 120 entity typings are agreed between two annotators. To achieve consensus, the annotators then collaboratively defined a set of rules to categorise the entities whose classes did not match and involved a third annotator for majority voting calculation. For example, an entity such as “List of Buffy the Vampire Slayer novels” was eventually classified as List instead of Novel, while the “Haute Aboujagane, New Brunswick”, which describes a specific community, was defined as Community instead of GovernmentAdministrativeRegion.

Figure 5.5 provides the overview on the composition of the resulting gold standard corpus with respect to general categories. “Place” and “Organisation” are the most present categories with respectively 20 and 16 entities in the dataset, corresponding to one of their child categories. For 18 entities that no appropriate category in the ontology could be found by the experts, those were labelled as “owl:Thing”.

<table>
<thead>
<tr>
<th>#Categories</th>
<th>20</th>
<th>18</th>
<th>16</th>
<th>10</th>
<th>9</th>
<th>7</th>
<th>5</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Categories</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Place</td>
<td>Company</td>
<td>PersonFunction</td>
<td>Device</td>
<td>UnitOfWork</td>
<td>Food</td>
<td>Species</td>
<td>ChemicalSubstance</td>
<td>Name</td>
<td>Activity</td>
<td>MeanOfTransportation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Medicine</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>EducationalInstitution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Event</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Language</td>
</tr>
</tbody>
</table>

**Figure 5.5:** Overview of the entity corpus

Experiments For each workflow, we assess their properties in the experiments with respect to quality, and cost of efforts. The accuracy of the predictor matching the gold standard will provide information about the predictor effectiveness for real applications. The accuracy of the human-based error correction with respect to the gold standard will provide insights into what quality can be expected from the crowd in general.

5.4.1 Prediction Quality

Intuitively, if looking at the predictor performance, it can be observed that $P_{auto}$ can only predict for 54 entities out of the 120 entities in the chosen dataset, while the other two human-based predictors could predict for all entities. In the case where $P_{auto}$ can not predict the category, it is considered as predicting the entity to be a “Thing”
which \( P_{\text{hybrid}} \) and \( P_{\text{naive}} \) could also assign to some entities based on the aggregated result from the crowd. Thus if simply looking at the accuracy of the predictor, it can be calculated as the percentage of entities that have been correctly classified, as shown in the formula below where \( \hat{N}_{Eo} \) represents the exact specific category predicted, and \( \text{Bernoulli}(l_{\text{gold}}, = \hat{N}_{Eo}) \) indicate the outcome (either 0 or 1) of comparing gold category with the category predicted by different predictor.

\[
\text{Accuracy} = \frac{\sum_{o} |O| \text{Bernoulli}(l_{\text{gold}}, = \hat{N}_{Eo})}{|O|}
\]

Thus, the accuracy based on the top predicted result for the exact specific category from \( P_{\text{auto}} \), \( P_{\text{hybrid}} \) and \( P_{\text{naive}} \) is 15.0%, 26.7%, and 32.5% respectively. The workflow based on \( P_{\text{auto}} \) and \( P_{\text{hybrid}} \) are designed to have further follow-up steps by human efforts to detect and correct errors, hence comparing accuracy of the predictor is not sufficient. This study thus also employs precision which measures the portion of correctly classified entities among all entities classified by the different workflow. The quality measures used are the precision-recall curves as well as the precision-recall break-even points for these curves. The precision-recall graph provides insights into how much precision it is possible to gain in case only a certain amount of recall is required. Typically, the head of the distribution achieves better precision due to the fact that the values correspond to the higher confidence of the predictor. The break-even point (BEP) is the precision/recall value at the point where precision equals recall, which is equal to the F1 measure, the harmonic mean of precision and recall, in that case. The results of the experiments for predictors described in Section 5.2.2 are shown in Figure 5.6. The main observations are:

The precision of the human computation based methods was generally higher when compared to the automatic method. Also, the automatic method did not produce any results for around 80% of the entities, whilst human computation based predictor provided suggestions for all entities with a certain quality. \( P_{\text{naive}} \) was showing the best precision over the test set of BEP 0.49, followed by \( P_{\text{hybrid}} \) with BEP 0.47. Surprisingly, the precision of \( P_{\text{naive}} \) was lower than expected for a completely human computation.
based solution and may indicate that the classification task requires high expertise and cannot rely on the crowd alone. For about 10\% of recall, all methods provide very good results, especially the precision for $P_{auto}$ and $P_{hybrid}$ was very high, making further steps in error detection and correction possibly unnecessary.

The confidence level of each predictor is standardised to the range $[0.0-1.0]$. The output quality at different confidence levels in terms of precision is plotted in Figure 5.7. As expected from previous results, high confidence levels above 0.9 for $P_{auto}$ and $P_{hybrid}$ indicate high quality results. For confidence levels below, the output cannot be trusted and need error correction. To improve the prediction, further research is needed to develop better predictors, or improve the quality of existing ones.

### 5.4.2 Cost

**Prediction costs** In comparison to $P_{auto}$, the predictors $P_{hybrid}$ and $P_{naive}$ required human input and therefore added additional effort costs. Whilst in $P_{hybrid}$ the user had to execute exactly one free text task per entity, to obtain the results using $P_{naive}$, the crowd worker had to complete 4.1 tasks on average. However, if added up the overall cost for getting a final predicted result from these three workflows, the completely human-based workflow $W_{naive}$ requires the least efforts cost as observed in table 5.1 while achieving higher accuracy than the other two.

**Error Detection Quality and Costs** Having received the output of a predictor, it tests whether the crowd was able to identify prediction errors and empirically determined the costs for this detection in terms of the number of questions needed to be answered.

<table>
<thead>
<tr>
<th></th>
<th>$W_{auto}$</th>
<th>$W_{naive}$</th>
<th>$W_{hybrid}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prediction costs</td>
<td>0</td>
<td>4.1</td>
<td>1</td>
</tr>
<tr>
<td>Error detection costs</td>
<td>2.9</td>
<td>0</td>
<td>1.6</td>
</tr>
<tr>
<td>Error correction costs</td>
<td>3.2</td>
<td>0</td>
<td>1.83</td>
</tr>
<tr>
<td>Sum</td>
<td>6.1</td>
<td>4.1</td>
<td>4.43</td>
</tr>
</tbody>
</table>

Table 5.1: Cost overview for different workflows.
on average. This study does not test error detection for \( P_{\text{naive}} \) as its output is already based on the crowd and cannot be expected to improve much with the error detection or correction step. As defined in the crowdsourcing task, at each step it shows maximum 7 candidate options to the user. On average, error detection took 2.9 detection steps with \( W_{\text{auto}} \) and 1.6 steps with \( W_{\text{hybrid}} \) as depicted in Table 5.1.

**Error Correction Quality and Costs** In the last step, it applies the algorithm to correct the errors produced by the predictors. Similar to the previous section, it measures the costs for the correction as the number of questions needed to be answered on average. The Table 5.1 provides an overview of the costs. \( W_{\text{auto}} \) required on average 3.2 steps to correct the prediction due to the fact that the predictor did not produce any results for the most entities and the whole tree had to be traversed to find the answer in such cases. 1.83 steps were required on average for \( W_{\text{hybrid}} \), indicating that in general, the predictor pointed the user to the right area within the ontology. In summary, as depicted in Table 5.1, \( W_{\text{auto}} \) appeared the most costly with 6.1 steps on average and the other two workflows showed comparable results.

### 5.4.3 The Quality of Human Output

Finally, it measures the quality of the workflows as a whole to estimate the effort to be invested by experts in post-processing. The Figure 5.8 shows the precision-recall curves of the human-based result correction for \( W_{\text{auto}} \) and \( W_{\text{hybrid}} \). It can be observed that in both workflows the result improved when compared to the prediction step alone. The proposed workflow \( W_{\text{hybrid}} \) reached a BEP of 0.53 - the highest result among all experiments.

---

**Figure 5.8:** The overall output quality of the workflows.
5.4.4 Crowdsourcing Tasks

This study decided to limit the number of top-options shown to the user to 7 as recommended in the literature. Longer lists may contain the correct result with higher probability, however, would also require more interaction and effort in complexity for a crowd worker. To show the possible influence of the option number on the prediction quality, the correspondence is plotted in Figure 5.9 where it varies the list size on the logarithmic scale from top 1 to the maximum of 740 possible categories and measure the predictor BEPs’ on the basis of the gold standards. As it is shown, the precision improves only slightly with the growing list size, indicating no meaningful advantage for lists with more than 7-10 options.

![Figure 5.9: Precision at different options list size with the W\textsubscript{hybrid} predictor](image)

5.5 Discussion

Each of the tested workflows has its merit and none of them seems to perform exceptionally well on the entities that the DBpedia Extraction Framework\textsuperscript{1} fails to classify. This section discusses these findings and their potential implications for the design of effective crowdsourced entity classification in DBpedia and beyond.

**Are unclassified entities unclassifiable?** As noted earlier, there were significant differences in the performance scores achieved in the experiments using different workflows. Especially notable is that the existing NLP tool could only classify 45% of the randomly selected entities (54 out of 120) with a precision of only 0.37. The human-based approach can achieve relatively higher 0.49 to 0.47 precision, but still a large portion of the entities are not classified to the correct specific type. To some extent, this indicates that untyped entities have certain characteristics which make them difficult to be classified.
• Firstly, it is observed that their types are quite diverse and not within the most popular classes\textsuperscript{14}. For instance, “Standard”, “SystemOfLaw”, or “TopicalConcept” are not categories non-expert could easily distinguish.

• Secondly, the imbalanced structure of DBpedia ontology also makes the classification of untyped entities whose boundary between subtle categories are not well defined. For example, “Hochosterwitz Castle” is a “Castle” which would be the most specific category for this entity, however, Castle is a child category of Building, which is a child type of ArchitecturalStructure that has many child types such as Arena, Venue, and Pyramid, leading the user to choose none of the children of “ArchitecturalStructure” as they did not see any fits. Similarly, “Place” and “Area” are both immediate first level types under “owl:Thing”, which create a lot of confusion in the first place as observed from the crowd contributed classification. Also, categories such as “Region”, “Locality” and “Settlement” are difficult to differentiate.

• Lastly, ambiguous entities unsurprisingly caused disagreement (Aroyo and Welty (2013b)). This was the case with “List” and specific types such as the “1993 in Film”,\textsuperscript{15} which is an List (not a film), and the “1976 NBA Finals”,\textsuperscript{16} which is rather a Tournament (child of “Event”, “SocietalEvent” and “SportsEvent”, but not a “List”). In general, entities like these contain a context which sometimes makes the entity itself ambiguous. In a similar way, “Provinces of the Dominican Republic”\textsuperscript{17} is a list (not a place) while “Luxembourg at the Olympics” is a sports team. In another case, an entity with context is just difficult to fit in any existing DBpedia types. For instance, “Higher education in Hong Kong” and “Petroleum industry in Nigeria”.

\textbf{The outputs are only as good as the inputs.} Taking naive workflow where maximum of 7 types (including a “NoneOfAbove” option) in one step is presented as an example, the aggregated outcome shows that 33 entities are categorised as “other” after traversing the DBpedia class tree top-down from “owl:Thing”, with none of the DBpedia categories being chosen. This also contributes to the ongoing debate in the crowdsourcing community regarding the use of miscellaneous categories (Aroyo and Welty (2013b); Feyisetan et al. (2015b)). In this case, using this option elicited a fair amount of information, even if it was used just to identify a problematic case. Feyisetan et al. (2015b) discuss the use of instructions as a means to help people complete an entity typing task for microblogs. In this case, however, performance enhancements would be best achieved by studying the nature of unclassified entities in more depth and looking for alternative workflows that do not involve automatic tools in cases which it assumes they will not be able to solve. One possible way to move forward would be to compile a list of

\textsuperscript{14}http://wiki.dbpedia.org/services-resources/datasets/data-set-39/data-set-statistics
\textsuperscript{15}https://en.wikipedia.org/wiki/1993_in_film
\textsuperscript{16}https://en.wikipedia.org/wiki/1976_NBA_Finals
\textsuperscript{17}https://en.wikipedia.org/wiki/Provinces_of_the_Dominican_Republic
entity types in the DBpedia ontology, which is notoriously difficult, and ask the crowd to comment upon that shortlist instead of the one more or less ‘guessed’ by a computer program. Another option would be to look at workflows that involve different types of crowds. However, it is worth mentioning that for the 120 randomly chosen untyped entities from DBpedia, 18 of them don’t fit in any DBpedia types based on the gold standard which indicate there is a need to enhance the ontology itself.

**Popular classes are not enough.** As noted earlier, entities which do not lend themselves easily to any form of automatic classification seem to be difficult to handle by humans as well. This is worrying, especially if we recall that this is precisely what people would expect crowd computing to excel at, enhancing the results of technology. However, we should also consider that a substantial share of microtask crowdsourcing applications addresses slightly different scenarios: (i) the crowd is either asked to perform the task on their own, in the absence of any algorithmically generated suggestions; or (ii) it is asked to create training data or to validate the results of an algorithm, under the assumption that those results will be meaningful to a large extent. The situation this study deals with here is fundamentally new because the machine part of the process does not work very well and distorts the wisdom of the crowds. These effects did not occur when using free annotations. An entity such as “Brunswick County North Carolina”[^18] is an obviously a “County” and a child type of “Place”. Freetext approach actually proposes this category, although that category does not exist in DBpedia yet. This result is consistent with Vickrey et al. (2008).

It became evident that in case the predicted categories are labelled in domain-specific or expert terminology, people tend not to select them. While under the unbound condition they are comfortable using differentiated categories, the vocabulary has to remain accessible. For example, Animal (a sub-category of Eukaryote) is used more than Eukaryote. In all three workflows, if the more general category is not listed, participants were inclined towards the more specialised option rather than higher-level themes such as Person, Place, and Location. This could be observed best in the freetext hybrid workflow. Such aspects could inform recent discussions in the DBpedia community towards a revision of the core ontology.

**Spam prevention.** It has been observed that crowdsourcing microtasks sometimes generate noisy data which either is submitted deliberately from lazy workers or from the crowd whose knowledge of the task area is not sufficient enough to meet certain accuracy criteria. The test question is a good way to help minimise the problems caused by both cases such that only the honest worker with basic understanding are involved in the tasks. The experiment in this study did not especially use control questions to prevent spam, instead, it uses test questions to recruit qualified workers. Although the test question approach requires about 10% additional judgments to be collected, it does give good inputs in which the definite spam is rare and negligible.

Using the Crowdsourced Data The validated and aggregated results may be lever-aged in several ways in the context of DBpedia. Freetext suggestions signal potential extensions of the DBpedia ontology (concepts and labels) and of the DBpedia Extraction Framework (mappings). Applying the freetext hybrid workflow gives insights into the limitations of entity typing technology, while the way people interact with the naïve workflow is interesting not only because it provides hints about the quality imbalances within the DBpedia ontology, but also for research on Semantic Web user interfaces and possible definition of new mapping rules for entity typing. In the same manner, other classification tasks such as image classification which has large number of categories or unfamiliar categories, present similar challenges in using automatic or hybrid predictor and might seek how to improve the native predictor by designing an effective multiple-step classification task.
Chapter 6

Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions

This chapter focuses on answering the second part of the RQ2, integrating the knowledge from previously investigated classification workflows and aggregation methods. It chooses the Snapshot Serengeti task, and improves the original design of classifying species with a further multiple-step classification. The result shows higher classification accuracy for the species. The implications of this work can be applied to similar classification tasks with a large number of categories.

6.1 Overview

This work investigates the second part of the RQ2 (when the classification involves a large number of options, whether the multiple-steps design can achieve a higher classification accuracy?). Following the “vision” of the ideal classification process in Figure
3.8, this thesis has investigated the classification component in which it proposed three alternative workflows and applied in an entity classification scenario (Bu et al., 2018). It has also focused on the multiple-step classification and the corresponding aggregation techniques, using image classifications from Zooniverse (Bu et al., 2019). In the real world, classification tasks with a large number of classification options (categories) are quite common, as seen in DBpedia entity classification and the species classification in Snapshot Serengeti\textsuperscript{15}. For such classification tasks, one could consider to utilise different workflows as proposed in the earlier chapter of this thesis: $W_{auto}$, $W_{naive}$ and $W_{hybrid}$ depending on the type of predictor used. On one hand, it has been observed that the completely human-based workflow $W_{naive}$ requires the least steps as observed in table 5.1 while achieving higher accuracy than the other two workflows. On the other hand, a classification task which has a large number of categories or unfamiliar categories, presents challenges in using automatic or hybrid predictor as facilitating tools are simply not available. Therefore, there is a need to investigate how to improve the native predictor by designing an effective multiple-step classification task to complete the classification.

It is noticeable that for a multiple-step classification, each classification step is like using a filter to help narrow down to a more specific group of options to some degree. To illustrate, for an entity classification, it might start classification from a generic category, then narrow down to a specific category. Similarly, in an image classification, each step might narrow down the given object to a group with a similar attribute or confirm the specific attribute value. However, it is observed in lots of classification tasks, the filter is provided as a facilitating feature which is optional to use, such as in Snapshot Serengeti\textsuperscript{15}. Existing work, including research and applications, mostly expose categories and options to users directly when the number of options is within an acceptable range (D. et al., 1979; Miller, 1956). When there are many options to choose from, a common best practice is to put those decision criteria into separate filters (see Figure 4.5) which are used as a facilitating mechanism to give users iteratively searching experience and immediate feedback on the narrowed down subsets of interest (Shneiderman, 1994; Heilbron and Niebles, 2014). Having seen different classification workflows and how they might impact the classification result, it is interesting to investigate this as a whole in a case where the same objects are being classified with different multiple-step task designs.

Snapshot Serengeti\textsuperscript{15} is a good candidate for such an experiment as it involves multiple steps and optional filters for the species classification step. Furthermore, during the experiment exploring aggregation methods in multiple-step classifications, we have saved the logs of user clicks from Snapshot Serengeti so that further insights might be obtained on how the filters have been used. This work is going to use Snapshot Serengeti as an example, validating under the same $W_{naive}$ workflow as before, using a further multiple-step design can help improve classification accuracy. Section 6.2 presents our hypothesis.
Section 6.3 explains the detailed design of the experiment. Results are presented in section 6.4, and the insights are discussed in section 6.5.

6.2 Hypothesis

The Snapshot Serengeti\textsuperscript{15} task’s aim is to classify images by identifying the species and the corresponding number of specific in the images. The original design of the task is a two-step classification (if there are animals present in the image): The first step is to identify the species from more than 50 given species with optional filters provided to help the users narrow down to a subset of species based on specific features; The second step is to count the number of the identified species in the image and choose from the given options of 1, 2, 3, ..., 10, 11-50 and 50+. Though some previous study, such as Yamauchi and Markman (1998), suggests dynamic filters which allow incremental and reversible changes result in high-level user satisfaction as it can give users feedback on the fly, it is not clear whether it impacts the resulting quality in classification tasks. Heilbron and Niebles (2014) does not present explicit filters, instead it uses binary filtering questions in the filtering stage when using the crowd worker for annotating web videos. The result shows that using the filtering strategy scales up with the number of videos, but in the two datasets experimented the dataset with more videos seems getting slightly decreased accuracy compared with the dataset with half number of the videos. To further the understanding, this study would investigate whether using mandatory questions (as filtering steps) would provide higher classification accuracy than providing option filters.

We hypothesize that in a classification task where an object needs to be classified into one category among a large number of categories, using optional filters and using compulsory filter-based questions would lead to different classification quality. Snapshot Serengeti task whose first question has more than 50 species (existing optional filters Likes, Pattern, Color, Horns, Tails) and the second question is to count the number of that species, is used as the baseline for the investigation. Our hypothesis can be summarized as following:

1. When having a lot of categories to choose from (more than 50 here in the Serengeti task), the use of filter-type questions can help improve the classification accuracy (for the species in the Serengeti task).

2. Number of the identified species is not affected by using the filter-questions that focus on species related characters.

The main idea is to convert existing filters in the original design into mandatory questions (Likes, Pattern, Color, Horns, Tails). Once all necessary filter-questions have been
Chapter 6 Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions

answered, the possible categories are listed for the user to final check, by comparing to the detailed example and corresponding species description. The subsequent count question remains the same as the original design. As to how to properly convert optional filters to filter questions, we need to consider both best practices from previous studies and the insights from filter usage logs collected from the baseline Snapshot Serengeti task. Section 6.3 elaborates these considerations and presents the detailed design.

6.3 Experiment Design

6.3.1 Design Considerations

Before we delve into the detailed design, it is worth examining how existing filters are used. Table 6.1 shows the filters/options and option groups from the previous version of Snapshot Serengeti\textsuperscript{15}. Figure 6.1 shows the number of species each specific filter could identify, which allows us to have a better idea of how to present the resulting subset of species when designing the task. Firstly, if using a single filter, more than half of these filters will still return a relatively big list (more than 15 candidates). Secondly, the filters that are able to narrow down the candidates to a small list (less than 10) almost all fall in “Horns”, “Looks Like” and “Build” filter group. However, these filter-wise statistics do not necessarily indicate those are the filters that should be first used for any objects that may not actually fall into those categories at all. Therefore, this study explores a user-driven process in which there will be no explicit optional filter(s) and users are able to choose which filter question to answer first.

<table>
<thead>
<tr>
<th>Filter (Filter group)</th>
<th>Number of options</th>
<th>Options (Sub-filters)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Like</td>
<td>6</td>
<td>['Cat/dog', 'Cow/horse', 'Antelope/deer', 'Bird', 'Weasel', 'Other']</td>
</tr>
<tr>
<td>Pattern</td>
<td>4</td>
<td>['Stripes', 'Bands', 'Spots', 'Solid']</td>
</tr>
<tr>
<td>Color</td>
<td>6</td>
<td>['Tan/yellow', 'Red', 'Brown', 'White', 'Gray', 'Black']</td>
</tr>
<tr>
<td>Horns</td>
<td>4</td>
<td>['Straight', 'Curved', 'Lyrate', 'Curly']</td>
</tr>
<tr>
<td>Tails</td>
<td>5</td>
<td>['Bushy', 'Smooth', 'Tufted', 'Long', 'Short']</td>
</tr>
<tr>
<td>Build</td>
<td>5</td>
<td>['Stocky', 'Tall', 'Lanky', 'Small', 'Low-slung']</td>
</tr>
</tbody>
</table>

Table 6.1: Filters and the corresponding options in Snapshot Serengeti

Furthermore, in the previous experiment of Serengeti task (4.3.3), we recorded the users’ clicks and the filters they used (though the use of a filter is optional). A quick analysis of the usage of these filters, along with the classification precision, presents an interesting picture (Figure 6.2 and 6.3). We could see the ones with more filters applied generally have higher precision. Additionally, the average number of filters used is mainly 1 or 2.
The main idea of the new task design is to convert the optional filters into mandatory classification questions while giving the user the freedom to choose which filter (question) to use (answer) to help facilitate their classification. The experiment design considers the best practices in cognitive science and UX design that are relevant to the classification task. The essential principle of the experiment design in this work follows from cognitive science research (Rosch and Lloyd, 1978; Miller, 1956; Nielsen and Loranger, 2006; D. et al., 1979; Proctor and Schneider, 2017). Existing user experience design principles such as Hick’s Law\(^1\) show that the number of stimuli/options affects individual reaction time and the more options to choose from, the longer it takes for the user to decide. The recent report from Proctor and Schneider (2017) presents various research that confirms that the number of categories, the way categories are organised and displayed (number of alternatives per screen), all have a role to play on top of Hick’s Law. Miller (1956) points out the memory limitation of an individual and has the “magical number 7” theory\(^2\) which has become a best practice in UX design. Nielsen and Loranger (2006) investigates how people interact with filters and sort searched results, indicating that only around 40% of people will look at the result rendered in a second screen or any information requiring scroll-down. As a result, when considering the number of options per step, this experiment should limit to a maximum of seven options.

\(^1\)https://en.wikipedia.org/wiki/Hick%27s_law

\(^2\)https://en.wikipedia.org/wiki/The_Magical_Number_Seven,_Plus_or_Minus_Two

**Figure 6.1:** Number of species filtered by a specific filter. Note: FilterGroup and Sub-Filter are the same as the ones shown in Table 6.1.
Figure 6.2: Species and the corresponding classification precision.
Figure 6.3: Correctly classified species and the corresponding number of filters used.
There are several aspects need to consider for the task design: First, if simply converting each available filter into a corresponding filter question, it requires considering the ordering of the questions. It could be based on the same order as it was shown on the filter tab of the existing baseline task (all filter options in the latest version of Snapshot Serengeti\(^3\) are picture-based for “looks like”\(^4\) but were text-based in previous version\(^5\)). Alternatively, it could consider putting the “easy” ones first which is tricky as the definition of easiness could vary: a) filters with fewer options are considered as easy. b) filters which have easy-to-distinguish options are considered easy. On the other hand, there are different ways how the options can be rendered, to name a few: a) displaying all candidates after each filtering operation. b) only showing candidates when possible choices are easy to be processed by the user. These factors are carefully considered during task design, guided by existing literature.

Considering the principles from existing literature, in this experiment, we want to give users the choice to choose what feature they feel most prominent or special in identifying species in the given image. Therefore, we show the filter groups as options for the classification question, users decide which feature (filter) they would like to identify from the image first. Only the relevant results will be shown for the final selection, or further feature filtering questions are prompted based on the number of candidates identified by the previous step. For the number of options to present on each step, it is based on the “magical number 7” theory\(^2\). There are also cases when the user needs to re-select or start over due to different reasons: one example would be the combined choices do not lead to a matching species (e.g. combination of “Color: gray, Horns: u-shaped, Tail: short” will result in “No matches”); another example would be the user is unsure of the choice just made and wants to re-check/re-select. For these cases, it is necessary to provide mechanisms for the user to go back to the last step or restart from the beginning. The detailed task is described in the following section 6.3.2.

6.3.2 Task

The classification task has the following logic:

- **Step 1:** By default, users will be shown all 6 filters (as features) and users need to choose one feature that they feel could most noticeable for the animal in the picture (Figure 6.4).

- **Step 2:** Corresponding options of the chosen filter/feature from step 1 will be shown (Figure 6.5). Users need to choose the value that best describes the animal under classification.

\(^3\)https://blog.snapshotserengeti.org/2018/01/05/snapshot-serengeti-upcoming-changes/
\(^4\)https://blog.snapshotserengeti.org/2018/01/31/snapshotsafari-vs-snapshot-serengeti-whats-changing/
\(^5\)https://blog.zooniverse.org/2012/12/11/snapshot-serengeti/
Chapter 6 Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions

125

Figure 6.4: Classification step 1 where existing optional filters are converted to options of a mandatory question

Figure 6.5: Example when horns filter/feature is chosen, corresponding options are shown

- Then, based on the result from step 2, different scenarios could occur:
  - If the number of the resulting candidates is less than the threshold number this study sets (it chooses 7 as the candidate threshold based on the best practice\(^2\)), candidates are displayed directly to the user. Figure 6.6 shows a screenshot of such a scenario. The example shows when choosing Horns, then choosing Curly, only three candidate species which is less than the threshold chosen, it will then directly show the list of species for the user to choose.
If the number of the resulting candidates from step 2 is more than 7, step 1 and step 2 is repeated, however, the major difference is the iterative step 1 (as shown in Figure 6.7) will only show the applicable filters for the resulting candidate sets. For example, if choosing Pattern and then Stripes, it still has 10 possible candidates which exceed the threshold this study sets. As a result, the program will show applicable filters to these candidate species which allows the user to further choose a feature to narrow down the scope. In this case, four filter options are applicable, ‘Looks like’, ‘Color’, ‘Tail’, and ‘Build’. If choosing color White, there will be three candidate species, which then can be shown to the user to choose from, similar to that in Figure 6.6. Note that even the resulting candidates from step 2 are more than 7, if there is only one applicable filter on the resulting candidate sets, it will automatically choose that filter asking the user to choose options from that filter.

It continues until the number of the resulting candidates is less than 7, or the user restarts the classification process.

Once a specific species is selected (Figure 6.8), questions asking for additional information such as the number of the identified species will be displayed, same as the existing Serengeti workflow.

Considering there is the case when the filtered candidates do not include the answer the user expects, it also includes the button to go back or restart which is the same as the existing Serengeti workflow. Figure 6.9 shows a screenshot with “Back” and “Restart” button which allow the user to easily navigate or restart the process as needed.

Once a user finishes the classification of one image, the same process is repeated for the next image, until all five images in the task unit have been classified. There is a progress indicator at the top of the task screen, as shown in Figure 6.4.
6.3.3 Task Setting

The classification task is deployed on the Figure-Eight platform\textsuperscript{2} (previously known as Crowdflower\textsuperscript{3}). Similar as before, based on the advice from the existing study (Acosta et al., 2013b), 5 units/images per task are configured. The worker completes a task by categorising five images.

**Judgment**: We choose to use 15 judgments per object throughout the experiment. It is the same setting as in the baseline Snapshot Serengeti task.

**Payment**: We pay 7 cents for each task consisting of 5 units. This setting considers the complexity of our classification task, providing comparable payment to existing work (Kittur et al., 2008; Ipeirotis, 2010; Acosta et al., 2013b).

**Quality control**: We do not use test questions for this experiment, as we want to compare the performance under the same condition with the baseline case in which the workers are not required to pass a test before working on the task.

6.3.4 Data

This study uses Snapshot Serengeti\textsuperscript{15} objects for this experiment. The previous data this study has collected consists of 4,149 objects with available gold standards. To evaluate the effect of using further multiple-step filter-based questions (via user-driver
Figure 6.8: Once a species is chosen, the description corresponding to that species is shown so that user can make sure this is the species they expect.

filter/feature selection), we could carry out the new design on a subset of these objects. Based on the power analysis technique\(^6\), it will need approximately two hundred objects to have a power of 0.80 assuming a 5% significance level and a two-sided test. As a result, this study has randomly selected 200 objects from 4149 objects.

\(^6\)http://www.statsoft.com/Textbook/Power-Analysis
Chapter 6 Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions

Figure 6.9: User is provided with Back/Restart in all steps (the example here is when the feature user has chosen does not match any species, they can go back to pick a different feature or restart the process)

<table>
<thead>
<tr>
<th>dataset</th>
<th>graph size</th>
<th>experiment</th>
<th>algorithm</th>
<th>accuracy per per level</th>
<th>accuracy by path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serengeti</td>
<td>54-11</td>
<td>Figure Eight crowd, new workflow</td>
<td>mv</td>
<td>species: 0.745</td>
<td>0.435</td>
</tr>
<tr>
<td>(200 random objects)</td>
<td></td>
<td></td>
<td></td>
<td>count: 0.515</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em</td>
<td>species: 0.645</td>
<td>0.365</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>count: 0.510</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp</td>
<td>species: 0.415</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>count: 0.360</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Figure Eight crowd, old workflow</td>
<td>mv</td>
<td>species: 0.725</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>count: 0.540</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>em</td>
<td>species: 0.570</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>count: 0.570</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mp</td>
<td>species: 0.270</td>
<td>0.23</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>count: 0.44</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.2: Classification accuracy of two different workflow: optional filter step (old) vs. mandatory filter question step (new)

6.4 Result

Table 6.2 shows the accuracy of each algorithm for the inferred answer for the Serengeti image classification. Comparing the classification accuracy for the “Species” in both the old workflow and new workflow, it is clear no matter which algorithm is used to infer the correct answer, the new workflow achieves higher classification accuracy (74.5% using mv, 64.5% using em, and 41.5% using mp). Considering the overall classification accuracy (by path), it could also see classification accuracy improves in the new workflow when inferred based on em or mp algorithm. However, inference based on mv does not seem to improve the classification accuracy if full path is considered, as the accuracy for classifying “Count” (number of animals) seems to be decreased a bit comparing to the old workflow in which by default only require one question (step) to be answered before answering the question of how many the identified species exists in the given image.
In terms of comparison amongst algorithms, \textit{mv} generally has higher accuracy for the Snapshot Serengeti classification case. For “Species”, \textit{mv} achieves the highest accuracy with 74.5\% and 72.5\% for new and old workflow respectively. \textit{em} comes the second with accuracy of 64.5\% and 57.0\% accordingly (as shown in Figure 6.10). A similar pattern is observed for the classification accuracy by path (Figure 6.11), \textit{mv} performs the best among three algorithms used, and \textit{mp} has the lowest classification accuracy.

![Accuracy by level (Species)](image)

\textbf{Figure 6.10: Accuracy by level (Species)}

Meanwhile, looking at the accuracy breakdown by node level, table 6.2 shows \textit{mv} has acceptable accuracy for the “Species”, but has poor accuracy (less than 55\%) for the “Count”, though it performs relatively better than other two algorithms tested. In particular, though \textit{mv} and \textit{em} obtained comparable accuracy (51.5\% and 51.0\%) on the “Count” level in the new workflow, the accuracy is notably lower compared to the old workflow. It may suggest that as it takes one or more clicks (steps) to identify the species, the user’s attention has been decreased by the time the “Count” question occurs, as suggested by previous research (Sweller, 1988; Van Merrienboer and Paas, 1998; Roda, 2011; Manresa-Yee et al., 2013) which indicate that at a given time user has limited attention to process presented information, and attention is vital to user’s performance.
Chapter 6 Combining Workflow Model and Aggregation Model: Multiple-step Classification with Filter Questions

Figure 6.11: Accuracy by path

<table>
<thead>
<tr>
<th>measure (sg sample 200)</th>
<th>mean</th>
<th>statistics</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>mv nodelevel 0</td>
<td>0.6949</td>
<td>-3.89607473737</td>
<td>0.000133723329768</td>
</tr>
<tr>
<td>em nodelevel 0</td>
<td>0.5753</td>
<td>-15.8761313845</td>
<td>5.40375456376e-37</td>
</tr>
<tr>
<td>mp nodelevel 0</td>
<td>0.32915</td>
<td>-26.3117333958</td>
<td>1.56972460929e-66</td>
</tr>
<tr>
<td>mv nodelevel 1</td>
<td>0.5149</td>
<td>4.33682000685</td>
<td>2.31279839081e-05</td>
</tr>
<tr>
<td>em nodelevel 1</td>
<td>0.52835</td>
<td>10.1095747864</td>
<td>1.65758902798e-19</td>
</tr>
<tr>
<td>mp nodelevel 1</td>
<td>0.3886</td>
<td>16.2231186631</td>
<td>4.68274367247e-38</td>
</tr>
<tr>
<td>mv</td>
<td>0.4265</td>
<td>1.88362737618</td>
<td>0.0610830406179</td>
</tr>
<tr>
<td>em</td>
<td>0.3468</td>
<td>-5.85340140451</td>
<td>2.04844542089e-08</td>
</tr>
<tr>
<td>mp</td>
<td>0.25775</td>
<td>-12.9711882788</td>
<td>3.49023243658e-28</td>
</tr>
</tbody>
</table>

Table 6.3: Significant testing result for comparing the accuracy of different algorithms between old and new workflow, based on node level (upper) or path (lower)

Significant testing was also performed, using the similar method described in section 4.2.4. The test randomly selected 50 times and each time with 100 objects (p-value threshold is set to 0.05 in code). All the numbers are significant, apart from the $mv$ (by path), as shown by the result logs in table 6.3. The fact that the accuracy by the label path by $mv$ is not improving seems to be consistent with the previous observation that $mv$ does not work that well when having more classification steps involved.
6.5 Discussion

Is using mandatory filter-based questions absolutely better? Though the experiment proves our hypothesis that using mandatory filter-based questions could produce higher accuracy for identifying the “Species”, it is not without side-effects. First, as it has been shown in section 6.4, the accuracy for the “Count” question that is asked after species has been identified seems to have decreased slightly. This was echoed by previous research on the cognitive load that shows users have a finite cognitive resource to process a task, and tend to lose attention (focus) once they got overload (such as the back and forth steps of choosing a different attribute to narrow down to a specific species). Second, based on the “Contribution Satisfaction” survey at the end of the task submission on Figure Eight, it shows users rate the “Ease of Job” for the new workflow (with mandatory filter-based questions to identify species) as 3.6 out of 5, compared to the 3.9 out of 5 for the old workflow (optional filters). This indicates that a user’s experience of having to choose some filter/attribute before identifying the species is not straightforward as intuitively choosing a filter, or directly choosing a species. However, considering the paid crowd context, introducing mandatory filter-based questions to narrow down to the expected options has the benefit of enforcing the seriousness of the user’s work. Kittur et al. (2008) investigates crowdsourcing micro-task design on Mechanical Turk and suggests that it is crucial to design the task in a way that answering the question both properly and randomly would require comparable efforts. In the case of optional filter versus mandatory filter-based questions for the species identification, the result from this study does show significant accuracy improvement.

Lessons for multiple-options classification with a large number of categories. Looking at the multiple-options cases explored in this research, objects such as DBpedia entity and Serengeti image present the challenge of having too many options (over 50 options for one classification question). In both cases, the “structure” of the options are known. For DBpedia, we know the ontology and the parent-child relationship between options. For identifying the species in Serengeti image, the filters/attributes and the species which has the corresponding attributes is known. The known structure makes it possible for converting a task with a question involving more than 50 options into manageable microtasks with fewer options. There are occasions where no additional information about the given options, does it mean extracting such filters/attributes related information is necessary? In the first place, it might not be a simple task to identify such features as it is normally domain-specific to the objects under classification. Moreover, if in any case, the task requesters identified a few features they are particularly interested, it might be worth using separate crowdsourcing task to classify a portion of the objects based on those features first, which then can be used as training data combining with expert identified gold standard for that portion of the objects. Furthermore,
dividing the options into groups and presenting fewer options to the user in a crowdsourcing task could be an alternative, as explored in the DBpedia entity classification case (particularly for the top-level parent categories).

**Potential of having one task with one type of questions?** The scenarios this study has been dealing with are multiple-step classification tasks that normally identify different interdependent aspects of the object under classification, and sometimes involves a large number of classification options for one or more classification questions. For instance, the SnapshotSerengeti scenario aims to identify the species (step 1) in the image and the corresponding number (step 2) of the identified species, the DarkSky case targets to identify the ones with cities in the picture (step 1) and the corresponding attributes of such picture: whether the picture is sharp (step 2) and sharp (step 3). To some degree, if the answers to two neighbouring classification questions are not tightly related (e.g. identifying the child category once parent category has been identified) and answering the first question might take considerable efforts (in terms of the information need to be processed, such as the case in this experiment of identifying the species), it might be potentially promising to putting the later classification question in a separate task to improve its classification accuracy. This should get the user’s full attention for the corresponding classification, without being affected by the cognitive overload or fatigue shown in previous studies (Sweller, 1988; Manresa-Yee et al., 2013).
Chapter 7

Conclusions and Future Work

This chapter summarises the contributions and concludes the research. It discusses the implications and limitations of this research, then lays out future research directions.

7.1 Conclusion

This section reviews our research questions, reflecting on what has been done to investigate them, what has been found and why it matters. It then concludes the research with design recommendations for multiple-step classifications.

Crowdsourcing has been popular for more than a decade since the term was first introduced (Howe, 2006), and has been widely applied in various contexts: writing (Bernstein et al., 2010), translation (Zaidan and Callison-Burch, 2011), emergency response (Zook et al., 2010; Ramchurn et al., 2015), traffic monitoring (Yan et al., 2009; Artikis et al., 2014), classification (Ho et al., 2013; Shamir et al., 2014), etc. In most of these applications, microtask crowdsourcing has been used. It decomposes a problem into smaller components making it possible to be tackled independently by non-expert users, which can help solve a wide range of problems that traditionally are expensive for experts to
do or difficult for a machine to handle. Researchers have investigated different aspects of microtask crowdsourcing, aiming to understand how to motivate the crowd, how to design, assign and execute a task in an effective way (e.g. assuring quality, optimising cost), and how to aggregate the collected data. One of the gaps this research identified is that for the more common multiple-step classification tasks, there is no work looking into it systematically. First, there is no work particularly looking into various multiple-step classification tasks trying to understand how task and workflow design could impact the quality of the classification results. Second, existing work of aggregation using various inference methods focuses on getting one single final result instead of looking at the choices (path) the crowd made along the multiple-step classification process. This research fills the gap by investigating both aspects and providing useful insights.

We have identified two research questions to navigate the investigation and created models to address the corresponding research questions. Our study presents a vision for an ideal crowdsourcing classification framework which contains three major components (Classification component, Aggregation Component, Quality Metric Component) and focuses on the first two components, to address the research questions. This thesis specifically focuses on multiple-step classification cases, and chooses to limit the scope in task-intrinsic design factors without considering factors such as incentives, crowd demographics/expertise, dynamic task allocation based on crowd performance, etc.

RQ1 focuses on the problem of Aggregation for true label inference and quality assessment in multiple-step classification. It conceptualises a graph model and proposes using the structure of a microtask crowdsourcing workflow as an additional feature to support inference algorithms in making decisions about correct labels, using data collected from the crowd. We have used three inference algorithms: majority voting, message passing, and expectation maximisation, and compared their performance using different aggregation strategies. The results have shown that no single algorithm consistently outperforms the others. While majority voting does well by metrics such as MARP and DCG, a deeper analysis of the accuracy of the produced candidate lists revealed a more nuanced picture. It also suggests the need for more dynamic inference approaches that can adapt to the complexity of the crowdsourcing workflow as well as the behaviour of the crowd. Overall, study of this research question contributes in two folds: it shows using different aggregation strategies (same or different algorithms) would affect the classification accuracy; additionally, it establishes the foundations for a possible aggregator that will utilise the results from individual algorithms to help achieve better performance at each node level or overall, for future research.

To address RQ2, this work starts with the Classification component and presents the “Classification Workflow Model” exploring workflow design in entity classification. In particular, we propose a crowdsourcing-based error detection and correction workflows for (semi-) automatic entity typing in DBpedia with a selection of the most specific category. Though the experiment employs DBpedia ontology, in principle it can also be
applied to other classification problems where the labels are structured as a tree. This work empirically evaluated the quality of the proposed predictors as well as the crowd performance and costs for each of the workflows using 120 DBpedia entities that are chosen uniformly at random from the set of entities, not yet annotated by the DBpedia community. The exploration shows promising performance when using the crowd as both the predictor and error detector/corrector. To further answer RQ2, this work then explores the effect of using optional filter and user-driven filter-based questions for the same set of objects classification that employs only humans in the process and involves multiple steps. The results show that compulsory filter question (as a step) does help improve the classification accuracy, compared with the scenario where there is no compulsory filter question, but a large number of categories are given, even though with optional filters as facilitating features. Findings from this experiment also show that the subsequent step of counting the number of the identified species has decreased accuracy compared with when having optional filter for species. This seems to suggest that the mandatory filter-based questions need more cognitive resources to process. It is probably a better choice just to keep a task focusing on a specific topic (answering a specific classification question with one or more facilitating questions). Overall, for a classification task, task requester could choose one of the three classification workflows, depending on the resources they have at hand. If there are no facilitation tools available, a completely human-based workflow ($W_{naive}$) is employed. In such a case, whether the task requester has some domain knowledge in the corresponding task area would affect whether they can use optional filters or mandatory questions to facilitate the classification if it involves a large number of options. The number of interdependent steps in a multiple-step classification would depend on how many different features of the object the task requester plans to collect, and might affect classification accuracy for some classification steps if classifications before them have consumed lots of attention from the user.

We conclude with following insights and design recommendations:

For a multiple-step classification task that aims to understand different aspects of the object being classified, it is necessary to know how to assess the quality of the classification results. Tasks like the CityAtNight, GalaxyZoo, and Snapshot Serengeti this work has investigated all involve more than one step to classify different aspects of the given object. The multiple steps are interdependent, but options in those steps do not necessarily have a parent-child relationship which is a specific type of multiple-step classification and is normally based on a hierarchical structure (e.g. DBpedia ontology). Therefore, only aggregating at single/last node level is not enough in a multiple-step classification task where options between classification steps do not have a parent-child relationship. At the same time, looking at how different aggregation methods could affect the results of multiple-step classifications helps us better understand the design dimensions of multiple-step classifications. The following presents the two main points:
• Our investigation shows the number of steps does affect the classification accuracy as observed in three Zooniverse multiple-step classification tasks. However, the difference between using paid-crowd or volunteer is negligible in GalaxyZoo case where more steps are involved. This indicates for multiple-step classifications similar to the GalaxyZoo case where though there are several steps, each step involves a reasonably small number of options, using volunteer and paid-crowd could achieve similar results.

• Furthermore, the accuracy in the multiple-step classification is also related to how the collected data is aggregated. The dependency-aware adapted approach this research proposed generally has higher classification accuracy. At the same time, among all three popular algorithms have been used in the adapted implementation in this research, majority voting seems to work consistently well in multiple-step classification aggregation. Expectation maximisation also performs quite well in most of the cases with dependency-aware adapted implementation. Message passing not only requires a longer time to run, but also highly depends on the sparseness of the data. It will produce poor classification accuracy especially when one of the steps involves too many options, or the task has more than three steps, and most of the crowd users only work on a few objects.

For a classification involving a large number of options (multiple-options), it is necessary to break-down into small questions to make the classification more solvable by the crowd users. Such tasks can be potentially turned into a multiple-step classification. This research investigates different ways for making the task relatively easier to be tackled by the crowd, ranging from leveraging alternative workflows (auto, hybrid, naive), to varying task designs under (naive) workflows with only human involvement in the classification steps:

• Facilitated by a predictor: This is the alternative workflow designs this research explored in handling the challenge of classifying an object into a large number of categories (Chapter 5), such as the one demonstrated in DBpedia entity classification. The workflow model this research proposed (predictor + error detector + error corrector) allows task requester to re-think how they want to implement the task, what automation tools or hybrid approaches they could utilise to provide a predicted option, which then present to the crowd to validate, or further classify to sub-categories identified. The variation of the alternative classification workflows ($W_{auto}$, $W_{hybrid}$ and $W_{naive}$) lies in the predictor stage which can be automatic, hybrid, or completely human-based ($P_{auto}$, $P_{hybrid}$ and $P_{naive}$). The experiment shows the completely human-based predictor has higher accuracy than the other two types of predictors. However, the other two workflows have their own merits: In particular, when an automatic tool is available in the corresponding
context with good predicting quality, it could save the time in identifying facilitating features or helping to navigate the crowd closer to the sub-category the object belongs to. Hybrid predictor, on the other hand, requires less error detection and correction when the predictor has effectively integrated initial human input with machine algorithms. It also shows the benefit of complementing existing categories the task requester already has but might not be sufficient, as it allows the task requester to seek freetext input which could suggest possible new categories.

- **Groups:** This might be the most straightforward approach of all three methods as it does not require task requester to have prior knowledge of additional features, or to have good automatic tools available or implemented hybrid process. It basically organises a large number of categories into smaller groups so it would not be overwhelming to the crowd worker in one go. However, it also involves variations that task requesters should consider before deciding how to group the categories to make more sense. A naive approach would be to present a small set of the categories in random order, as seen in Chapter 5. Such naive grouping does not need to consider whether the categories put into the same group have any similarities or not, neither does it consider the order of the categories. It would be suitable to use this naive grouping when the task requester doesn’t have further information, or it might be too complicated to organise it in a more effective way. Other grouping strategies focus on further reducing the cognitive load on the crowd worker, such as whether to put the easier categories first, present in alphabetic order, or group categories by a certain degree of similarity. This is considered in the work in Chapter 6, when deciding how to present the filter-based questions (each question will present a corresponding group of categories), to make it easier to the user. One of the choices, as shown in this work is a user-driven process where we give the choice to the user to decide which feature (related with the filter-based question) they want to identify first. It has been proven with significant accuracy improvement in the Serengeti species case. Task requester could consider using groups to reduce the cognitive efforts for the crowd users when they have too many categories, like the one in Serengeti project with more than 50 species, or the one in the DBpedia ontology with more than 50 categories in the first level and more than 600 specific categories.

- **Filters or filter-based questions:** Task requesters normally provide a facilitating filter to help the crowd workers to narrow down their choices, as it is seen in the original Snapshot Serengeti project. The way the filter is provided varies as shown in this research: either as optional filters (Chapter 4) or turned into mandatory filter-based questions (Chapter 6). The latter one does show significant classification accuracy improvement when applied in the Snapshot Serengeti classification case which has more than fifty options (species the crowd workers need to identify from the given image). It is possible to use such mechanism only when the task
requesters have good knowledge about the features of the target options (categories) and can navigate the crowd to potential candidate options once the crowd identified certain features from the given object.

This research clearly demonstrates that for classifying a specific attribute of an object that involves a large number of options, such as classifying species, converting it to a multiple-step classification will significantly improve the classification accuracy, but it also raises the question of whether it affects the classification accuracy of other steps when such classification is already part of a multiple-step classification which has subsequent steps (such as classifying the number of the identified species). This indicates that it might be better the task answers single-topic question(s), as it shows the accuracy of the classification step afterward got slightly decreased. Focusing on one single topic does not mean one single question, instead, it emphasises the steps/questions in a task should focus on answering a specific question such as identifying the species – it could go through multiple-steps (e.g. filter based questions to narrow down species). Then have a separate task asking for counting the number of the identified species from the given object (image). However, it will accordingly raise the needs of recruiting more crowd workers and might also cost more (from time and/or money aspect). This is something we did not dig further in this thesis and is further discussed in the following section.

7.2 Discussion

As we have seen from this research, it is hard to design a task to achieve high classification accuracy because the whole process involves so many factors. Even though this research focuses on a specific area – task and workflow intrinsic factors (without considering motivation, gamification, task difficulty/complexity, dynamic task allocation, etc), and the aggregation strategy for answer inference in multiple-step classification, it presents many variables.

**Single-best aggregation strategy and algorithm?** The investigation on aggregation in the multiple-step classification does seem to present a nuanced picture though in most cases mv perform consistently well. It shows no single algorithm performs best on all scenarios amongst all three algorithms this research used (mv, mp and em). Due to time and resource limitation, we did not compare with some of the popular machine learning algorithms which might require a part of the data to have true labels (classified by experts) first. It is also interesting to see how the number of steps has affected the accuracy under naive and adapted aggregation, using individual algorithms. The aggregation strategy this research employs is not sophisticated but could produce reasonably well results. Nonetheless, there are other strategies which might produce better results, such as a heuristic aggregator utilising results from several algorithms. As the main aim
of this thesis is to find a way to help us assess the quality in multiple-step classification, a reasonably well-performed strategy is good enough for us to use. Consequently, we did not spend more time on other strategies which could be one of our future work.

**Is hybrid or automatic approach better than pure human-based workflow?**

This thesis explores three workflows that are based on complete human efforts ($W_{naive}$), human efforts and processing tools ($W_{hybrid}$), and the automatic prediction tool ($W_{auto}$). Using DBpedia entity classification as an example, this research shows the human-based approach produces a higher quality result. It also shows that automatic and hybrid approach seem to cost more efforts, not to mention the tool selection needed for automatic/hybrid approach in itself is a challenging task. However, due to time constraints, this was only tested with entity classification cases where automatic entity typing tool is available. In lots of other cases, $W_{auto}$ might not be possible, and $W_{hybrid}$ is still tricky as the algorithm or tool choose to process the freetext data in itself is also challenging and can be difficult to find for certain domains. It needs further investigation to understand how hybrid and automatic workflow can be devised for a wider range of classification tasks. Therefore, it is hard to conclude which workflow is better as it is case-by-case: some tasks may have domain-specific tools to facilitate the classifications, while others don’t; even when a certain tool might be available, the accuracy of the tool needs to be considered to avoid too much error detection and correction efforts in the later process.

**Multiple-options in a single-step or multiple-step classification?**

Multiple-step classification covers multiple interdependent questions which might be a simple question with binary options, or question with multiple options. The multiple-option case such as Snapshot Serengeti species identification from an image, or the DBpedia entity classification from more than 50 top-level options (parent categories), requires different treatment to make it less overwhelming to the user. Strategies for using groupings, filters, filter-based questions, have been explored in this research. This research was not able to cover further investigation due to time constraints. It is worth comparing the performance between random grouping and other strategies, such as popular group presenting first, or simply presenting by alphabetical order, or separating filter-based questions for classifying a specific topic (such as species) from questions focusing on other topics (such as counting the number of species). It would be better such design factors are tested on more than one dataset, not only to understand how well these can be generalised, but also to better understand how subsequent classification steps might be affected if preceded by steps which consume lots of cognitive resources.

All these implications are crucial for researchers and task requesters to be aware of. Most importantly, they are the areas researchers should look further into in the future.
7.3 Future Direction

This work has focused on crowdsourcing classification tasks and explored the impact task and workflow design could have on the classification quality. It limits the scope only to task intrinsic features during the study. Along the way, due to the needs of assessing the quality of multiple-step classification which has not been fully studied by existing research, the first research question it deals with is to explore a few algorithms and understand different aggregation strategies to help achieve higher accuracy (mainly provide insight on how these algorithms help in inferring true label in such multiple-step classification, as gold standards are not always available). However, the study has its limitation in both the task/workflow design factors explored, and the inference methods can be applied. This section looks at the limitations and suggests corresponding future research.

**Aggregation in multiple-step classification** To assess results from multiple-step classification tasks, this work modelled the scenarios as a graph-searching problem and utilised existing popular individual algorithms to find the true path in the classification structure based DAG. Though both EM and MP have the advantage of being able to take into account the user performance, in most of the classification tasks, the fact that not all workers contribute on all objects, leads to two limitations: on one hand, many workers may have only worked on a few objects (regardless of a large number of objects needed to be classified) and hence the worker response is sparse; on the other hand, it is less possible users agree on the same label path compared with that users agree on the same node. As a result, the object response is even more sparse when the full label path was treated as an atomic label. With such limitation in mind, the straightforward run-on-path approach is likely to perform poorly compared with the dependency-aware adapted strategy utilising relatively “dense” labels for individual nodes. From our point of view, the strength of each of these individual algorithms could be better leveraged, in an ensembling manner (e.g. a heuristic method using multiple algorithms with different weights). Though such an ensemble method seems promising, the thresholds/weights used in this case need further investigations in order to get the optimal results. It might also be interesting to look at other advanced algorithms, such as the ones that can incorporate object difficulty, or the ones that are network-based and are suitable for inference in graphs.

**Task and workflow related factors** Classification quality can be affected by many reasons as presented in the background (Chapter 2). Factors such as user incentives, or workers’ specific expertise level, are not considered in our research. This study is limited to the case where the structure (classification categories) are given, and how different workflow (individual steps are designed and chained together) can lead to better classification accuracy. Thus, the focus is more of the depth and width of the classification structure, corresponding to the number of steps and options involved in each step.
the task design factors, there are many aspects concerning the user interface and user experiences, such as layout, rendering style, timely feedback or tooltip, these are not something this work has particularly investigated. It is worth further investigations on these areas for multiple-step classifications. For workflow design, this thesis investigated three workflows using entity classification for which some existing automation tools can be utilised. Further efforts to understand how hybrid and automatic workflow can be devised for a wider range of classification tasks are needed, in order to save time and cost while achieving high quality.

Datasets In this study, it studied both entity classification and image classification cases. Though we investigated the workflows using entity classification as a use case, the alternative workflows proposed could be applied in both cases. Nonetheless, this thesis only has a limited number of datasets explored, one entity classification with DBpedia and three image classification cases with Citizen Science projects. The suggestions derived from this research is aiming for general multiple-step classifications, it would have been better to test this on more datasets involving more diverse tasks.
Bibliography


Yoram Bachrach, Tom Minka, and John Guiver. How To Grade a Test Without Knowing the Answers — A Bayesian Graphical Model for Adaptive Crowdsourcing and Aptitude Testing. 2012b.


Sebastian Deterding, Dan Dixon, Rilla Khaled, and Lennart Nacke. From game design elements to gamefulness: defining "gamification", 2011.


Ujwal Gadiraju, Jie Yang, and Alessandro Bozzon. Clarity is a Worthwhile Quality. pages 5–14, 2017.


Jonathon S Hare, Maribel Acosta, Anna Weston, Elena Simperl, Sina Samangoei, David Dupplaw, and Paul H Lewis. An Investigation of Techniques that Aim to


Harmanpreet Kaur, Alex C Williams, Anne Loomis Thompson, Walter S Lasecki, Shamsi T Iqbal, and Jaime Teevan. Creating Better Action Plans for Writing Tasks via Vocabulary-Based Planning. 2(November), 2018.

Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. Worker types and personality traits in crowdsourcing relevance labels. (October):1941, 2011.


Aniket Kittur, Jeffrey V. Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In


Fenglong Ma, Yaliang Li, Qi Li, Minghui Qui, Jing Gao, Shi Zhi, Bo Zhao, Heng Ji, and Jiawei Han. FaitCrowd: Fine Grained Truth Discovery for Crowdsourced Data Aggregation. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 745–754, 2015.


George A Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.


Long Tran-Thanh, Trung Dong Huynh, Avi Rosenfeld, Sarvapali Ramchurn, and Nicholas Jennings. *Crowdsourcing complex workflows under budget constraints*, 2015.


Honglei Zhuang, Aditya Parameswaran, Dan Roth, and Jiawei Han. Debiasing Crowdsourced Batches. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1593–1602, 2015.
