Selective classification for logistics

DISSERTATION

to obtain the degree of Doctor
at the Open Universiteit
on the authority of the Rector Magnificus,
prof. dr. Th.J. Bastiaens
before the Committee appointed by
the Doctorate Board
to defend in public

on Thursday, December 19, 2024 in Heerlen
at 4.00 PM exactly

by
Xi Chen
Born on 13 April 1992 in Jiang Xi, China

Selective classification for logistics

PROEFSCHRIFT

ter verkrijging van de graad van doctor
aan de Open Universiteit
op gezag van de rector magnificus
prof. dr. Th.J. Bastiaens
ten overstaan van een door het
College voor promoties ingestelde commissie
in het openbaar te verdedigen

op donderdag 19 december 2024 te Heerlen
om 16.00 uur precies

door
Xi Chen
geboren op 13 April 1992 te Jiang Xi, China

Promotores:
Professor Dr. T.E.J. Vos Open Universiteit
Dr. S. Bromuri, Open Universiteit

Copromotor:
Dr. D.S. Tan, Open Universiteit

Leden beoordelingscommissie:
Professor Dr. A. Gatt, University of Utrecht
Professor Dr. N. Alechina, Open Universiteit
Professor Dr. R.W. Helms, Open Universiteit
Professor Dr. H.P.E. Vranken, Open Universiteit
Professor Dr. M. Luštrek, Jožef Stefan Institute

Abstract

Machine learning is widely employed in our daily lives. However, the underlying models often fail to meet expectations, which can be caused by imperfect datasets or the inherent complexity of the problem itself. Classification models, for example, will always make predictions even in situations of uncertainty. This can lead to costly errors in critical areas such as autonomous driving, medical diagnosis, and logistics.

Selective classification offers a solution by enabling models to reject samples when the model is uncertain, thereby enhancing performance and safety by avoiding costly mistakes.

This thesis contributes to the advancement of selective classification methods, showcasing their applicability in the logistics industry by developing real-world applications and providing insights on data issues. Additionally, it offers a process for tackling industrial problems, which is important for further research and implementation in other industries.

We begin by identifying two key challenges in the logistics domain: predicting Harmonized System codes and importers, and demonstrating the effectiveness of selective classification. We then propose an improved rejection criterion, the ‘confidence range,’ to enhance classifier performance. Recognizing the importance of data quality in logistics, we introduce a transfer learning approach that uses a confidence score to assess whether the data contains sufficient information for classification. Lastly, we present a novel approach to modeling shipment journeys as a language model. This allows us to determine how to selectively update customers based on their shipment’s progress.

Abstract(Dutch)

Machine learning wordt veelvuldig toegepast in ons dagelijks leven. De onderliggende modellen voldoen echter vaak niet aan de verwachtingen, wat kan worden veroorzaakt door onvolmaakte datasets of de inherente complexiteit van het probleem zelf. Classificatiemodellen, bijvoorbeeld, zullen altijd voorspellingen doen, zelfs in situaties van onzekerheid. Dit kan leiden tot kostbare fouten in kritieke gebieden zoals autonoom rijden, medische diagnoses en logistiek.

Selectieve classificatie biedt een oplossing door modellen in staat te stellen om samples af te wijzen wanneer het model onzeker is, waardoor de prestaties en veiligheid worden verbeterd door kostbare fouten te vermijden.

Deze thesis draagt bij aan de vooruitgang van selectieve classificatiemethoden, waarbij de toepasbaarheid ervan in de logistieke sector wordt aangetoond door het ontwikkelen van toepassingen en door het bieden van inzichten over data-issues. Daarnaast biedt het een proces voor het aanpakken van industriële problemen, wat belangrijk is voor verder onderzoek en implementatie in andere sectoren.

We beginnen met het identificeren van twee belangrijke uitdagingen in het logistieke domein: het voorspellen van Geharmoniseerde Systeemcodes en importeurs, en het aantonen van de effectiviteit van selectieve classificatie. Vervolgens stellen we een verbeterd afwijzingscriterium voor, het 'vertrouwensbereik,' om de prestaties van de classifier te verbeteren. Met erkenning van het belang van datakwaliteit in de logistiek, introduceren we een transfer learning-benadering die gebruikmaakt van een betrouwbaarheidsscore om te beoordelen of de data voldoende informatie bevat voor classificatie. Ten slotte presenteren we een nieuwe benadering voor het modelleren van verzendingsreizen als een taalmodel. Dit stelt ons in staat om te bepalen hoe we klanten selectief kunnen updaten op basis van de voortgang van hun zending.

Proposition

1. The existing terminology does not accurately capture the diversity in selective classification.
2. Choose your baseline model carefully, as the rejection approach may perform differently across various models.
3. Usability is of significant importance in the industry.
4. A pre-defined rejection score presents a challenge for the model to learn; conversely, a model finds it easier to learn from a flexible target.
5. Data quality represents an unavoidable and critical issue within the industry.
6. Language models can be applied to more than just text-based tasks.
7. The emergence of Large Language Models (LLMs) has significantly shaped the direction of most text-based research.
8. Research becomes more engaging when its findings can be immediately applied.
9. In the industry, understanding how to frame a real-world issue as a data science problem is often more critical than the modeling aspect itself.
10. Life is not a sprint; it's a marathon. Remember to take breaks from time to time.

Acknowledgment

This journey towards completing my PhD has been one of the most challenging and rewarding experiences of my life. It was a path filled with sleepless nights, moments of self-doubt, and being haunted by impostor syndrome. I feel more relief than accomplishment as it all comes to an end. I am very thankful to the many people who helped make this journey not just possible, but also enjoyable at times.

To Stefano and Daniel, your support was critical in crossing the finish line. I truly believe I couldn't have completed this journey without your help. Your guidance and encouragement made all the difference. Also, thanks to Tanja and colleagues at OU for their support, supervision, and valuable feedback on my work.

A special acknowledgment goes to my parents, who provides selfless support in their own way.

I also extend my thanks to Thomas who provided this opportunity, and the support from all the lovely people in the Data Analytics Team at DHL. The experiences and challenges encountered during my time with you guys have been invaluable to my personal and professional growth.

To my friends: Kevin, Shuangqi, Tianyi, Sofia, Max, Inaki, Yunhai, and everyone I met during this journey who has kept me mentally sane, thank you.

I would like to quote one of the sentence from Haruki Murakami in the end: "Pain is inevitable, suffering is optional". As we journey through life, may we all find the courage to face the challenges.

Contents

Abstract **i**

Abstract(Dutch) **iii**

1 Introduction **1**
1.1 Selective classification 2
1.2 Confidence score criteria 3
1.3 Different types of selective classification 5
1.4 Selective classification in logistics 7
1.5 Methodology 10
1.6 Author contribution and outline 12

2 Background **15**
2.1 Definition 15
2.2 Rejection methodologies 16
2.3 Summary 28

3 Neural Machine Translation for Harmonized System Codes Prediction **29**
3.1 Introduction 30
3.2 Related Work 31
3.3 Data 33
3.4 Methods 34
3.5 Results 37
3.6 Conclusion 38

4 Pair-wise selective classification for shipment importer prediction **39**
4.1 Introduction 40
4.2 Related work 41
4.3 Methodology 44
4.4 Experiment 48

4.5 Conclusion 51

5 Confidence Range: Bridging Failure Detection and True Class Probability on selective hierarchical text classification 53
5.1 Introduction 54
5.2 Related work 56
5.3 Methods 58
5.4 Experiments 63
5.5 Conclusion 69

6 A transfer learning approach to predict shipment description quality 71
6.1 Introduction 72
6.2 Related work 72
6.3 Data 74
6.4 Methodology 75
6.5 Results and Discussion 77
6.6 Conclusion 78

7 Language Modeling in Logistics: Customer Calling Prediction 79
7.1 Introduction 80
7.2 Method 81
7.3 Experiment 83
7.4 Results & Discussion 83
7.5 Conclusion & Future work 85

8 Conclusions 87
8.1 Research Question 87
8.2 Considerations about Industrial deployment 90
8.3 Considerations about LLMs 92
8.4 Societal Impact 92

Bibliography 97
References 97

Appendix 113

List of Figures 113

List of Tables 115

Chapter 1

Introduction

The field of machine learning (ML) has witnessed a remarkable transformation over the past few decades, evolving from rudimentary algorithms to sophisticated models capable of performing complex tasks across various domains.

Neural Networks (NNs), a specific type of machine learning, are inspired by the architecture of the human brain. They utilize interconnected neurons organized into layers to analyze various forms of data, allowing for the automatic extraction of intricate features from raw data and facilitating tasks such as image and speech recognition. With the surge in computational power and data volume, deep neural networks (DNNs) have emerged. These networks have deep layers of neurons, enabling them to learn intricate patterns and representations from data more effectively compared to NNs.

NNs and DNNs have undergone rapid and substantial evolution in recent years. They have manifested remarkable achievements across a wide array of domains [RBL+22; VSP+17; RKX+23; CLB+21; SLL+23], demonstrating their versatile applicability and innovative potential. Moreover, DNNs are now emerging as a prominent feature in our daily lives [SBE+21; MA20], which subtly transform the way we interact with technology and each other.

However, these models still encounter challenges when deployed in diverse scenarios. A primary concern is the model's lack of a "do not know" response mechanism. Classification models, for instance, will always make a prediction, even under circumstances of uncertainty. This is particularly problematic in areas with long-tail distributions, such as autonomous driving, logistics, or medical diagnosis, where it is impractical to anticipate every possible scenario during the model training phase. Another issue, which is more common in reality, is when the model performance fails to meet the business or regulatory requirements, particularly in industrial contexts [JOK+12], where data quality often falls significantly short of ideal standards. The cost of the mistakes is relatively high [BDD+16] and can even cause severe irreversible consequences.

In these cases, selective classification is crucial due to its ability to selectively identify which inputs they can handle reliably and which ones they are uncertain

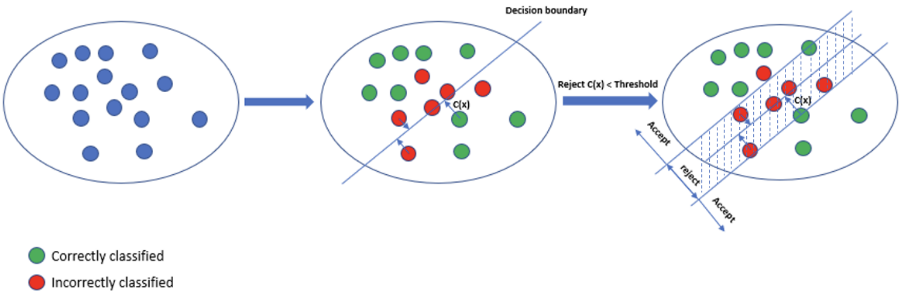


Figure 1.1: Selective classification example: Using the distance to decision boundary to reject samples. The confidence score $C(x)$ here is the distance to the decision boundary. Without any rejection, the model can achieve 60% precision with 100% recall. However, if we apply the rejection as shown in the figure, the model achieves 87.5% precision with 53.3% recall.

about. This allows it to pass uncertain cases to a human instead of risking a costly mistake, thereby improving the safety and reliability of the model.

1.1 Selective classification

In a standard classification process, the classification model assigns an input into one of the predefined classes that the model was trained on. The problem arises when we give it an input that does not belong to predefined classes. For example, we can train a classification model that distinguishes between cats and dogs, but then a bird appears. The classification model will be forced to classify the bird as either a cat or a dog since these are the only two classes that the model knows, and both of which are wrong. This example might seem bizarre, but it happens in real-world domains such as in self-driving cars where novel objects can randomly appear on the streets, or in the logistics domain where a new importer's product needs to be cleared by Customs.

Selective classification incorporates an additional step: the capability to reject specific samples on which the model is prone to errors or unsure in prediction. As illustrated in Figure 1.1, a simple implementation of this approach is to reject samples based on their proximity to the decision boundary. This assumes that points closer to the decision boundary are the confusing examples that the model is uncertain of. By rejecting these confusing examples, the precision of the model on the processed data is enhanced.

The term 'selective classification' was introduced by El-Yaniv [El-$^{+}$10].

This concept, however, is not entirely new. The foundational ideas of 'Reject Option' and the 'Error-Reject Trade-off' were discussed as early as the 1970s [Cho70], underpinning what would later be recognized as selective classification. By adjusting the rejection threshold distance, as illustrated in Figure 1.1, it is possible to generate a precision-recall plot, depicted in Figure 1.2. The appropriate threshold can be selected based on the specific requirements of the problem.

Another commonly used evaluation metric is the risk-coverage curve, where risk is defined as the cross-entropy loss for a classification task [GE19]. From a business perspective, the risk does not directly reflect the business value or requirements. Instead, businesses typically prioritize precision. Specifically, they aim to maximize the number of items processed (recall) while meeting specific precision requirements. Therefore, in this thesis, we use precision-recall curves to evaluate the performance of the models.

For multi-class classification problems, we use micro-average precision and recall [SL09] considering the problems discussed here have hundreds to millions of different classes. Also, the individual class is not important for the analysis. The micro-average precision/recall is defined as follows:

$$\text{Micro-averaged precision/recall} = \frac{\sum_{i=1}^{k} \text{True Positives}_i}{\sum_{i=1}^{k} (\text{True Positives}_i + \text{False Positives}_i)} \tag{1.1}$$

where k is the number of classes. The score that is used for rejection is denoted by $C(x)$, it is referred to by various names, such as a confidence score, model confidence, uncertainty, or trust score, depending on its method of inference.

To maintain clarity and avoid confusion, this thesis will use the term 'confidence score' for $C(x)$ irrespective of the inference method. In Figure 1.1, $C(x)$ is represented by the distance to the decision boundary. The objective of the selective classifier is to reject as many misclassified (or high-risk) samples while accepting as many correctly classified samples. Consequently, misclassified (or high-risk) samples should have lower confidence scores, whereas correctly classified (or low-risk) samples should have higher confidence scores.

1.2 Confidence score criteria

There are two groups of methodologies to derive the confidence score $C(x)$: One is establishing a learnable or pre-existing target to act as the confidence score. The other one is explicitly modeling the source of uncertainty.

The difficulty of the first methodology lies in defining an ideal learnable/existing confidence target that meets two criteria: 1. Higher confidence score

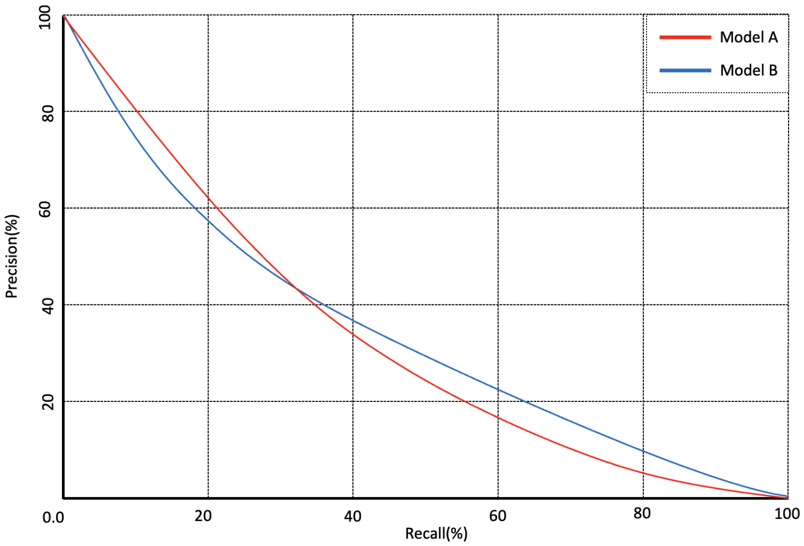


Figure 1.2: Precision-recall plot: The tradeoff between between precision and recall performance of the model. A better rejection approach should have higher recall under the same precision. Model A performs better before the intersection point while Model B performs better after the intersection point.

correlates with higher accuracy. 2. The model can generalize the confidence score target well. Our experiments show that the previous state-of-the-art predicted true class probability (TCP) [CTB+19] does not generalize well in hierarchical text classification (HTC) problem (discussed in Section 5). Our contribution is to introduce a flexible confidence score target that the model can learn to predict on new samples. More specifically, a new confidence score called confidence range (CR) is proposed. The CR combines the idea of failure detection, which indicates how likely the model is to make a mistake [HDV18] and the TCP score that represents the probability of the true class, as explained in Chapter 5.

The target for the first methodology could be represented in several ways, such as the highest softmax output from the classifier [GE17], the ratio derived from the nearest neighbor trust score [JKG+18], a failure detection score indicating the likelihood of misclassification [HDV18], or the probability of the ground truth label [CTB+19], among others. These confidence scores effectively rank samples based on their likelihood of being correctly classified, without considering the specific source of potential errors. The details of those approaches are explained in Chapter 2.2.

The second methodology, which focuses on explicitly modeling uncertainty, is categorized into data uncertainty (aleatoric) and model uncertainty (epistemic) as shown in Figure 1.3. This methodology is gaining popularity due to its expressiveness. It looks at confidence by dividing it into two distinct components:

1. **Data uncertainty**: This aspect concerns aleatoric uncertainty, reflecting the noise in the measurements or inherent randomness in the system being modeled. It represents the variability in the outcome that cannot be reduced even if we had more data.
2. **Model uncertainty**: This refers to epistemic or model uncertainty, which is the uncertainty in the model parameters themselves. This type of uncertainty arises due to a lack of knowledge or information and is theoretically reducible as we collect more data or improve our models [HW21].

Both types of uncertainties can be used as confidence score in selective classification [CZG20; LSS20; LLP$^+$20]. The challenges in explicitly modeling the different types of uncertainty include:

1. Separating aleatoric and epistemic uncertainty proves challenging.
2. Determining the best approach to model this uncertainty remains an open question in ongoing research.

In this thesis, we did not directly enhance the methodology itself; instead, we aimed to utilize these criteria to better understand complex data in logistics, such as text and time series data. Chapter 7 demonstrates how these criteria can be employed to understand data quality.

1.3 Different types of selective classification

There are different ways to group selective classifications: model-agnostic versus model-specific and cost-known versus cost-unknown.

Model-agnostic vs model-specific. In the earlier stages of machine learning, before neural networks became predominant, the rejection approaches were largely model-specific. This period saw the development of tailored approaches for tree-based models [KW06; Hel70], SVMs [GRK$^+$08]. These model-specific approaches, while effective within their respective frameworks, posed significant challenges in terms of adaptability and transferability to other models.

As machine learning evolved, there was a shift towards deep learning, drawing increased attention to developing models capable of handling more complex data and tasks. This shift also influenced the domain of selective

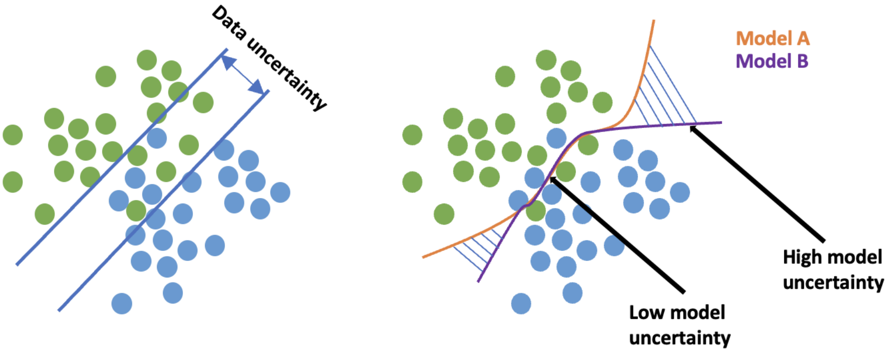


Figure 1.3: A toy example of aleatoric and epistemic uncertainty.

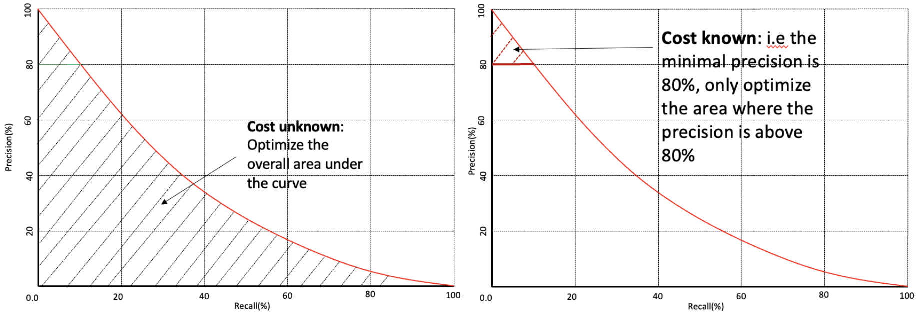


Figure 1.4: Optimization difference between cost-known vs cost-unknown.

classification, with a growing preference for deep learning-based approaches. Despite the diversity in deep learning architectures, the recent methodologies are easy to transfer across different neural networks with minimal adjustments.

This thesis primarily focuses on developing selective classification approaches that are not tied to any specific model architecture.

Cost-known vs cost-unknown. In the realm of selective classification, the concept of ‘cost-unknown’ refers to scenarios where the cost associated with misclassifying a sample is not predetermined, or where a minimum precision level is not defined. This scenario mirrors the realities faced by many applications where optimization of model performance across the precision-recall curve as shown in Figure 1.4. The advantage of this approach is its provision of a comprehensive overview of model performance across varying precision thresholds.

Conversely, ‘cost-known’ means the cost of misclassified samples can be quantified, or a minimum precision level is predefined. This allows for the translation of cost-known situations into a subset of cost-unknown scenarios where the optimization is only focusing on a specific precision range.

In the logistics industry, the requirements for model performance exhibit significant variation, driven by distinct preferences and regulatory standards across different countries and departments. This diversity often results in a ‘cost-unknown’ situation at the outset of many projects, where the specific costs associated with misclassification are not predefined. Acknowledging this prevalent challenge, our research is focused on the ‘cost-unknown’ domain.

There is also a cost associated with rejection as it requires humans to classify it manually. This is use case specific, it will be discussed within each problem we solved here.

1.4 Selective classification in logistics

Major logistics corporations, including DHL and FedEx, operate across multiple departments, each offering substantial opportunities for the integration of machine learning to enhance operational efficiency and service quality. These applications span a wide range of logistics functions, such as package transportation—including pickup and delivery processes, warehouse management, scheduling [HXZ+21], customs clearance [CBV21], and routing optimization [HWC+20]. The deployment of machine learning in these areas not only promises to streamline operations but also to significantly elevate the precision and responsiveness of these logistical tasks.

There are numerous support functions available, including customer service and financial services. Notably, these areas possess a large amount of domain-specific data that is challenging to acquire through other channels. Additionally, the volume of shipments handled by top-tier logistics companies is substantial, exceeding one million per month. Enhancing automation or increasing the rate of automation can significantly impact efficiency improvements.

In this thesis, our primary focus is on improving the quality of the clearance process and customer service, based on an initial evaluation of the potential high financial impact.

1.4.1 Clearance process

The clearance process encompasses all the necessary steps required to comply with international trade regulations and customs formalities, ensuring that goods are legally and efficiently transported from one country to another.

The clearance process stands as a critical component, ensuring the seamless movement of goods across borders and through various regulatory landscapes.

Logistics companies play a pivotal role in this complex process, acting as intermediaries between shippers, carriers, and regulatory bodies to facilitate the efficient and compliant transportation of goods.

The logistics company needs to submit valid shipment-related information. This include the shipper, receiver, importer, exporter, shipment description, dimension, and harmonized system codes (HS Codes): A standardized number used to classify products for international trade.

HS Codes and importers are not always known by the customer. It requires the broker or the agent in a logistics company to manually classify it based on the description provided by the customer, the complexity of the hierarchical structure of the HS Code (chapter, heading, subheading) makes it rather difficult for an agent to classify.

Submitting incorrect shipment-related information to Customs will lead to the rejection of the clearance process, which will put the shipment on hold until the correct information is submitted. This naturally might lead to delays in the shipment due to missing the truck/flight connection. More severely, it might lead to the revocation of the clearance license if the accuracy of the HS Codes declaration is too low.

Ensuring the accuracy of HS Codes and importer classifications is essential for logistics companies. Currently, agents classify HS Codes and importers manually because the accuracy of the HS Codes classification model and importer prediction model does not meet business requirements, as discussed in Chapters 3, 4, and 6. Therefore, having a rejection mechanism like selective classification is necessary to reject the unsure or likely misclassified samples, to ensure that the auto-classified samples can meet compliance. Considering the large volume shipped through the logistic company, being able to auto-classify part of the shipments can have a significant impact.

Ensuring high-quality descriptions is also essential. Generic or incomplete descriptions that lack sufficient information cannot reliably identify HS Codes. Therefore, a mechanism to assess description quality is equally important.

Our work in this area is:

- Validate the existing rejection methodology in HS Codes and importer prediction problem (Chapter 3 and 4).

- Develop a better rejection methodology (Confidence range) that can increase the automation rate in HS Codes prediction problem (Chapter 5).

- Use confidence score to identify shipment description quality (Chapter 6).

1.4.2 Customer service

Customer service plays a crucial role in establishing trust, ensuring satisfaction, and maintaining long-term relationships with clients. In logistics, customer service encompasses various elements including timely and accurate communication, problem-solving, tracking and reporting, and providing customized solutions.

In total, the customer service department in a large logistic company such as DHL receives over half a million calls per month regarding shipment status, representing a significant portion of their daily workload. To enhance efficiency and customer satisfaction, we first need to understand why customers are calling. Secondly, we explore whether we can predict when a customer will call and provide updates before they reach out to the call center.

A language model in logistics is proposed to address customer call prediction, as shown in Chapter 7. Due to the randomness of customer behavior, we need to approach customers who are most likely to call. Additionally, given that the shipment journey represents complex time-series data, we incorporate uncertainty as one of the rejection criteria to better understand the data.

Our work in this area is:

- Propose a language model that can simulate the shipment journey. Such a model can be used in different downstream tasks.
- Validate the language model in the customer calling prediction problem, which enables the customer center to selectively update customers who are likely to call.
- Utilize model uncertainty analysis to understand data label noise.

1.4.3 Research question

The objective of this thesis is to develop a better confidence score $C(x)$ for selective classification and to investigate further how to utilize it to evaluate the data quality. We chose the hierarchical text classification problem to study due to its significant impact on the HS Codes classification problem: automating the HS Codes classification process could save large logistics companies millions of euros annually by partially automating what is currently a manual classification task performed by agents.

Moreover, the design of the rejection methodology can be dependent on the architectural design of the deep learning model. Therefore, our first research question focuses on identifying the ideal base model for hierarchical text classification. To enhance the automation rate, which is directly linked to the potential savings in effort, we began to examine data error patterns and consider how to design a more effective confidence score to improve the automation rate while meeting specific performance requirements. Additionally, data issues present a serious challenge in the industry. To understand the limitations, we explored the use of different uncertainties to gain insights into data quality issues. In summary, our research questions are as follows:

- What deep learning models are suitable for hierarchical text classification?
- How to use the error pattern to design a better confidence score?
- How to use the confidence score to analyze data quality (1. Whether the data contains enough information for the category. 2. Model uncertainty analysis to understand data label noise.)?

By addressing these questions, this research seeks to offer tangible improvements in the automation processes of large logistics operations.

For the first question, Chapters 3 and 5 looked into existing HTC methodologies and validated different models, establishing the Transformer-based seq2seq model as highly effective for HTC challenges. This finding is significant as it directs logistics companies towards adopting this model for improved classification accuracy and efficiency, leading to potential cost savings and enhanced operational workflows.

Addressing the second question, we validate the concept of rejection scoring in Chapter 4 by evaluating an existing approach in logistics, setting the stage for our contribution in Chapter 5. Here, we introduce a new rejection criterion, the “confidence range” which leverages a combination of model error analysis and predicted probability on the ground truth label. This criterion enhances the automation rate even further.

For the third question, our research extends into evaluating data quality through advanced techniques as described in Chapters 6 and 7. We demonstrate how a confidence score, derived from transfer learning, can effectively assess data quality as discussed in Chapter 6. Additionally, distinguishing whether errors originate from the data or the model provides crucial insights into the limitations faced when dealing with complex data problems. This distinction is not merely academic; it has practical implications for the entire machine learning pipeline as Chapter 7 shows.

1.5 Methodology

In a logistics company, problems or challenges are typically proposed by the business or identified by data analysts and scientists from the data. Business-proposed problems often address process bottlenecks or pain points in daily operations, such as the HS Codes classification problem, where the company needs to hire numerous domain experts to classify a large volume of shipments due to policy changes. Alternatively, problems can be identified by data scientists or analysts. For example, this has happened when the shipment description quality had to be evaluted, where quality issues in the descriptions became evident in the data. These problems undergo an evaluation process and are ranked based on difficulty, potential benefits, and stakeholder engagement. Problems with high potential for success and significant benefits will be brought to the discussion with stakeholders or higher managers.

Once a problem is approved by stakeholders, the data science team typically follows the Cross-Industry Standard Process for Data Mining (CRISP-DM) [WH00] methodology. Such a methodology, as shown in Figure 1.5, is a widely adopted framework for structuring data mining projects and it comprises six phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. This methodology has been chosen, at the early onset of the data analytics group in which the author is embedded, because it ensures alignment with business objectives, promotes an iterative approach for continuous improvement, and facilitates collaboration with the stakeholders.

Among the identified problems, three problems tackled in this thesis ranked among the highest-valued projects: (1) HS Codes prediction, (2) importer prediction, and (3) customer calling prediction. For these problems, we detail each specific process (except their deployment in the production environment) in Chapters 3, 4 and 7 respectively.

Chapters 5 and 6 is the extension of the HS Codes prediction problem discussed in chapter 3. We noticed the accuracy of the HS Codes classification model could not meet the initially defined business requirements. Hence, selective classification was used to meet such requirements on the part of the data in which the prediction model had high confidence. The work in Chapter 6 focuses on deepening business and data understanding to identify the limitations in the data and process behind the collection of shipment descriptions. It later proposes a better model, which is designed in Chapter 5 during a second iteration of a CRISP-DM cycle. In this cycle, the business understanding is re-evaluated based on the results obtained, using selective classification analysis to define the part of the data where automation is possible due to higher confidence in certain data segments.

With respect to CRISP-DM, the thesis focuses on the stages between business understanding, modeling, and evaluation, whereas specific aspects concerning the deployment of the algorithms in production are left out of the scope of this thesis. This is due to the fact that the thesis happened in an industrial context in which the concerns were divided between analysis of the problem and technical implementation of the solution in a production system, with the author's work focusing on modeling, training, and evaluating machine

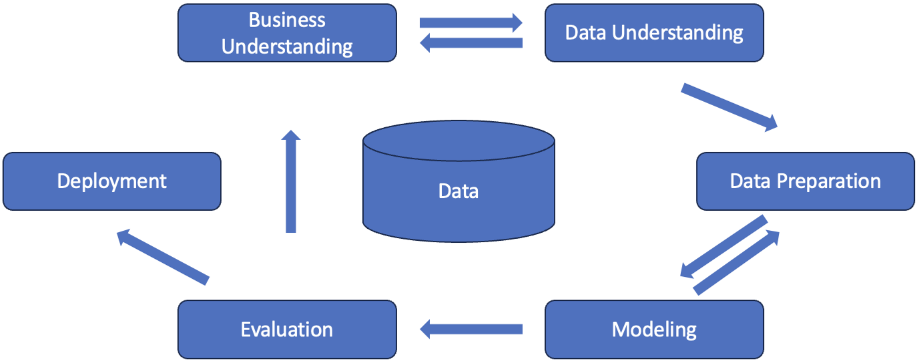


Figure 1.5: CRISP-DM methodology

learning models, with the purpose of delivering the trained models to a software engineering unit dedicated to deployment tasks. Despite such a division of concerns, insights from the deployment of the proposed solutions in Chapter 3 and Chapter 6 have influenced the research proposed in this thesis by creating new business understanding, which is then taken into account when defining the approach described in Chapter 5 and the modeling of the applications in Chapter 4 and Chapter 7.

1.6 Author contribution and outline

1.6.1 Author Contribution

This thesis focuses on selective classification in the logistics domain. From an application perspective, it demonstrates how selective classification can be applied in logistics by addressing multiple real-life problems: HS Codes prediction, importer prediction, and customer calling prediction. Additionally, we propose simulating the shipment journey as a language model and using uncertainty to understand the data. It can inspire other industries to adopt selective classification. From the perspective of scientific advancement, this work enhances the state of the art by introducing a new confidence score: the confidence range in the HS Codes classification problem. Furthermore, it looks into how to utilize confidence score to understand data quality. First, it proposes a method for utilizing a confidence score to evaluate shipment description quality when labeled data is limited. This approach can be extended to use cases such as assessing the quality of feedback and reviews.

1.6.2 Thesis Outline

Chapter 2 presents a background concerning selective classification, it offers a novel perspective on the methodologies used in selective classification over the past five years. The contribution to the field is that we categorize the existing approaches based on their confidence score methods, rather than the methodologies claimed by their authors. It provides a clearer understanding of the landscape of selective classification techniques, enabling a more intuitive comparison and assessment of their effectiveness.

Chapters 3, 4, 5 (Under review), 6 and 7 are reported from the publications [CBV21; CBT22; CTG+23; CBV22; CAT+23].

Chapter 3 introduces an application of neural machine translation architecture to the classification of HS Codes, an important logistic problem involving hierarchical text classification within the selective classification context. It also proposes a new hierarchical loss function tailored to this classification challenge. This work was published at ICMLT (ICMLT 2021: 2021 6th International Conference on Machine Learning Technologies, Jeju Island, Republic of Korea, April 2021).

Chapter 4 focuses on the logistics clearance process. This chapter proposes an alternative classification approach with a rejection option, aimed at automating a significant portion of the shipment process. It also examines the effectiveness of the TCP approach in this context, with findings published at ICMLC (ICMLC '23: Proceedings of the 2023 15th International Conference on Machine Learning and Computing), showcasing the potential for operational improvements in logistics through machine learning.

Chapter 5 building on the foundations laid in the previous chapter, this chapter addresses limitations in existing approaches to failure detection and true class probability. It introduces a novel concept: the confidence range, which synergizes both aspects to enhance performance in hierarchical text classification tasks. This work is currently under review by Applied Intelligence, a journal published by Springer.

Chapter 6 explores the application of confidence score in assessing the quality of shipment description data, addressing a gap in the industry for a standardized evaluation method. The discussion centers on how confidence score derived from transfer learning can serve as an important feature for data quality assessment. This work was published at SAC '22 (SAC '22: Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing).

The final research chapter 7 focuses on the application of selective classification in language modeling to logistics, treating the shipment journey as a sequence of events that can be modeled linguistically. The trained language model, tested in customer calling prediction use case, allows the company to selectively update customers who are likely to call. Also, in order to better understand the performance, we utilized the state-of-the-art uncertainty measurement to understand where the limitations come from. This work was published at ESANN 2023 (ESANN 2023 - European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning).

Chapter 8 provides a summary of the thesis, including a discussion of future work and its potential impact.

Chapter 2

Background

In the field of selective classification research, several review papers are available. Zhang et al. [ZXL$^+$23] provide valuable insights into general rejection mechanisms, categorizing them into confidence, calibration, and discrimination. However, these works have not fully explored the connection between rejection scores and model uncertainty. Similarly, dedicated studies on uncertainty, such as those by Hüllermeier et al. [HW21], Gawlikowski et al. [GTA$^+$23], and Abdar et al. [APH$^+$21], offer in-depth discussions on uncertainty but do not address it in the context of selective classification.

This chapter provides a background on selective classification. To bring better clarity within this domain, we have organized the background based on the underlying methodological approaches.

The definitions and evaluation metrics will be presented in the beginning. Subsequently, each methodology will be discussed.

2.1 Definition

In the following we will assume the training data $S_n = \{(x_i,\ y_i)\}_{i=0}^{n}$ is sampled independent and identically distributed from some unknown underlying distribution $P(X, Y)$. A selective classifier [GE17] is a pair of functions (f, g), where $f : X \to Y$ is a classifier that predicts class labels given an input, and $g : X \to \{0, 1\}$ is a selection function that determines whether to reject the prediction or not. The selective classifier is defined as follows:

$$(f, g)(x) = \begin{cases} f(x), & \text{if } g(x) = 1 \\ \text{reject}, & \text{if } g(x) = 0 \end{cases} \tag{2.1}$$

The selection function $g(x)$ is dependent on a confidence score $C(x)$ and a threshold τ, defined as follows:

$$g(x) = \begin{cases} 1, & \text{if } C(x) \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{2.2}$$

Various evaluation metrics have been proposed to evaluate selective classification. Risk-coverage [El-+10; GE17] is one of the early evaluation metrics to measure the performance of the selective classifier, where the risk is defined as the average loss on the accepted samples:

$$R(f,g) \triangleq \frac{E_P[\ell(f(x),y)g(x)]}{\phi(f,g)} \tag{2.3}$$

where $E_P[\ell(f(x),y)g(x)]$ is the expected risk, $\ell(f(x),y)$ is a loss function, $\phi(f,g)$ is the coverage.

The issue with the risk-coverage plot is that it does not directly reflect the business value (e.g. automation rate), as a low risk does not guarantee better precision. Also, it does not give any insights into the percentage of the samples that can be classified. Thus when it comes to industry usage, precision-recall [MS99; DG06; HG16] is better used in the context of selection classification. That is also why we use the precision-recall metrics in our work as Figure 1.2 shows.

2.2 Rejection methodologies

The rejection methodology is categorized as shown in Figure 2.1, which includes four main categories. Additionally, we separately discuss a fifth category that is not depicted in the figure.

The first category includes the softmax baseline and its associated regularization techniques. This approach is widely used in selective classification due to its convenience and effectiveness. The idea is to use the softmax output as a confidence score to determine whether to reject or retain the prediction. The second category, the nearest neighbor approach, is well-known for its explainability, a crucial attribute in sensitive industries such as law; however, it is less commonly used due to scalability challenges. Here, the confidence score depends on the proximity to the nearest neighbor. The third category focuses on uncertainty, assessing it from model, data, or prediction perspectives. In this case, the confidence score can reflect data uncertainty, model uncertainty, or predictive uncertainty. The fourth category ranks samples based on their likelihood of being correctly or incorrectly classified; this likelihood can be a learnable target or an aggregate score obtained during training. The probability of correct classification can be treated as the confidence score. Finally, the last category is the model ensemble, which is not shown in Figur 2.1. It uses the ensembled softmax output to represent the confidence score.

The approaches that do not use precision-recall or similar evaluation matrics will not included here. Thus, in addition to the aforementioned models, there exist calibration approaches that address the overconfidence issue in deep learn-

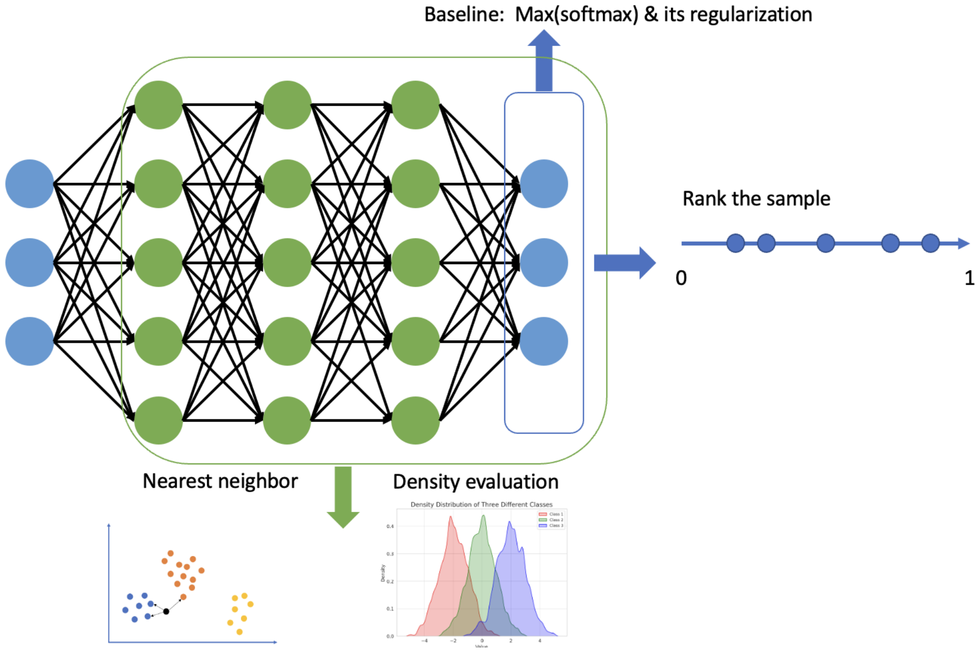


Figure 2.1: Different types of confidence score: softmax and its variations, nearest neighbor, density evaluation, and ranking. The baseline and its regularization use the maximal value of the softmax to represent the confidence score. The nearest neighbor based approaches use the distance or the amount of nearest neighbor to represent its confidence in hidden layer(s) or last layer. The density-based approaches calculate the density score in its training data to represent the confidence score, and the ranking-based approaches use different targets that are derived from training data/process to represent the confidence score.

Table 2.1: Summary of Softmax-related approaches

Paper	Year	Approach
Softmax[HG16][GE17]	2017	Baseline
Confidence-penalty & Label smoothing [PTC+17]	2017	Regularization
Temperature scaling & perpatration [LLS18]	2018	Data augmentation & Regularization
Relaxed softmax [NZV18]	2018	Regularization
Mixup & Smoothing [TCB+19]	2019	Data augmentation & Regularization
Reg mixup [PYL+22]	2022	Data augmentation
LogitNorm [WXC+22]	2022	Regularization
FMFP [ZCZ+22]	2022	Regularization
DOCTOR [GRG+21]	2022	Regularization
Umix [HLY+22]	2022	Data augementation
Openmix [ZCZ+23]	2023	Data augementation

ing models [GPS+17; JOK+12]. We will not discuss these in detail here because those approaches aim to align confidence with accuracy. This differs from the objective of selective classification and can even reduce prediction-recall performance [ZCZ+22]. Also, we will not discuss the Bayesian models as they are barely evaluated under the context of selective classification [TAS18; HL20; MNP21]. Also, those approaches are not scalable for industry applications due to expensive computational costs.

2.2.1 Softmax baseline

In neural networks, it has been proven that the maximal value of the softmax layer is a reliable indicator for identifying misclassified and out-of-distribution examples [HG16; HG16; GE17], making it a strong baseline for selective classification due to its high performance and ease of use [GE17]. However, a significant issue highlighted with the maximal softmax probability in deep neural networks is its tendency to be overconfident, even in some misclassified samples [NYC15; GPS+17], which leads to very confident mistakes. To address this issue, various approaches have been proposed. Based on their methodology, they can be grouped into two categories: regularization approaches and data augmentation. The summary is shown in Table 2.1.

Inspired by principles of reinforcement learning, Pereyra et al. [PTC+17] introduced output regularization techniques to mitigate overconfidence in deep neural networks. They proposed two specific regularizers: a confidence penalty based on maximizing entropy and label smoothing. Their findings demonstrate that penalizing low entropy in output distributions serves as an effective mechanism for regularizing deep neural networks, contributing significantly to the ongoing efforts to enhance model reliability.

Building on the work of Pereyra et al. [PTC+17], Liang et al. [LLS18] extended these concepts by introducing two additional techniques aimed at model calibration and robustness. The first technique, temperature scaling, involves a post-processing calibration of the network through an adjusted softmax function, mathematically represented as $Softmax(logit/T)$, where T denotes the temperature parameter. The second technique focuses on enhancing model resilience against adversarial attacks by introducing small, controlled perturbations to the inputs, effectively improving the model's ability to distinguish between in-distribution and out-of-distribution data.

Neumann et al. [NZV18] further contributed to this line of research by developing a relaxed softmax approach, an advanced version of temperature scaling that allows for sample-based temperature adjustments rather than a fixed global parameter. This enables more granular control over the regularization process.

Thulasidasan et al. [TCB+19] validated the mix-up training, a method originally proposed by Zhang et al. [ZCD+18]. This technique involves the creation of virtual training samples by blending pairs of random samples drawn from the training dataset (x_i, y_i) and (x_j, y_j), using a mixing parameter λ. The new training sample (x'_i, y'_i) is:

$$\begin{aligned} x'_i &= \lambda x_i + (1-\lambda)x_j \\ y'_i &= \lambda y_i + (1-\lambda)y_j \end{aligned}$$

where $\lambda \in [0, 1]$

The mix-up training approach has been shown to significantly enhance model robustness, further contributing to the development of more reliable and resilient deep learning models.

Along with the mixup approach, Pinto et al. [PYL+22] introduced RegMixup, a novel technique that synergizes empirical data distribution approximations with the Mixup approach to enhance training data diversity. This method aims to refine the training process by balancing between the original data and Mixup-generated data.

Following a similar trajectory, Han et al. [HLY+22] and Zhu et al. [ZCZ+23] presented variations of the mix-up approach that further expand its application. Han et al. [HLY+22] developed UMix, which integrates a weighted linear combination of the original and mix-up losses. This adjustment aims to prioritize samples with lower performance, encouraging the model to allocate more attention to these potentially underrepresented data points. Zhu et al. [ZCZ+23] proposed OpenMix, which leverages outliers to enrich the training

Table 2.2: Summary of nearest neighbor approaches

Paper	Year	Layer	Distance/Density Function
Distance-based confidence score [MAN17]	2017	single hidden layer	K-nearest-neighbor
Deep KNN [PM18]	2018	all hidden layers	Ensemble on nearest neighbor
Trust score [JKG+18]	2018	last layer	Distance ratio
Deep Weighted Averaging Classifiers [CZS19]	2019	modified last layer	weighted nearest neighbor
Justification-Based Reliability [VIY20]	2020	all layers	

dataset and introduces an additional reject class for the model to predict, thereby enhancing the model's robustness and its ability to handle out-of-distribution data.

Zhu et al. [ZCZ+22] critically assessed the effectiveness of various regularization approaches in enhancing failure detection capabilities of DNNs. Contrary to common strategies, they advocated for the use of stochastic weight averaging, as proposed by Izmailov et al. [IPG+18], to navigate towards flatter minima, arguing that this approach significantly improves the model's generalization and reliability.

Further exploring indicators for misclassification detection, Granese et al. [GRG+21] identified the ℓ_2-norm of the softmax output as a promising metric. However, Xia et al. [XB22] observed that this behavior is similar to maximal softmax response, suggesting that while useful, it may not provide substantially new insights into misclassification detection beyond existing methodologies.

2.2.2 Nearest neighbor approach

The nearest neighbor approach is famous for its explainability, a crucial aspect in sectors such as banking and law enforcement, where the clarity of decision-making processes is more crucial than model accuracy [BGR+99; Abd15]. This characteristic underlines the importance of explainable models in sensitive applications.

The hidden layers in deep learning models offer a robust representation of input features [CYK+18; LZM22], making them suitable candidates for integration with the k-nearest neighbors (KNN) algorithm. Mandelbaum et al. [MAN17] enhanced this integration by introducing a confidence score derived from the local density estimation of points within an embedded space formed by a trained network. This method involves using the Euclidean distance between a point and its k nearest neighbors in the embedded space of the training set to measure local density. To facilitate effective embedding, Mandelbaum et al. [MAN17] suggested two strategies: 1. minimizing distance for samples with identical labels and maximizing it for those with different labels. 2. Adversarial training.

Instead of looking at the neighbor on a single layer level, Papernot et al. [PM18] proposed the Deep KNN—a method that calculates the nearest neighbors across all hidden layers of a DNN. This approach not only leverages the deep learning model's layered representations but also enhances resilience against adversarial inputs.

Instead of applying KNN in all of the hidden layers as Papernot et al. [PM18] did, Virani et al. [VIY20] trained a nearest neighbor search tree based on the information from training data and distance metric for inference purposes. This method categorizes outcomes into three levels of certainty: known, potentially known, and unknown directly instead of using a threshold on a rejection score.

Card et al. [CZS19] introduced an alternative to traditional classification approaches by implementing a deep weighted average method, which replaces the softmax layer with a weighted sum of the nearest neighbor across the labels of instances in the corresponding training set, using a Gaussian kernel based on Euclidean distance for weighing.

The choice of which layer's output to use for feature representation remains a topic of debate [PGK$^+$11; HZW$^+$16]. In this context, Jiang et al. [JKG$^+$18] proposed a trust score that leverages the output of the last layer. This score uses the ratio between the distance to the nearest class and the distance to another class that is different from the predicted class. This approach supports the notion that trust scores derived from the deeper, more processed layers of a DNN tend to offer more reliable indicators of the model's predictions.

2.2.3 Uncertainty

The uncertainty in the model's prediction arises from two sources: the uncertainty inherent to the model (epistemic uncertainty) and the uncertainty intrinsic to the data itself (aleatoric uncertainty). Epistemic uncertainty, generally stemming from lacking of the knowledge, can be mitigated by increasing the training dataset [Gal16; DD09]. Conversely, aleatoric uncertainty originates from the noise present in the data generation process and can not be reduced by increasing more training data. Understanding and decomposing predictive uncertainty into these components allow for a better interpretation of model predictions and aid in making decisions based on the reliability of these predictions.

In the realm of deep learning, Gaussian Process (GP) is commonly used in modeling the uncertainty. It models the entire distribution of possible functions that could describe the data.

Given a set of input-output pairs (X, Y), where X represents the input

Table 2.3: Summary of single uncertainty-related approaches

Paper	Year	Layer	Density function
GPDNN [BMG17]	2017	Modified last layer	GP
Evidential deep learning [SKK18]	2018	last layer	Dirichlet distribution
Dirichlet Prior Network [MG18]	2018	last layer	Dirichlet distribution
DUQ [VST$^+$20]	2020	Modified last layer	RBF network
SNGP [LLP$^+$20]	2020	Modified last layer	GP with RBF kernels
DUE [vASJ$^+$21]	2021	Modified last layer	GP with RBF kernels

features and Y represents the corresponding output values, a GP can be defined by its mean function $m(x)$ and covariance function $k(x, x')$. For simplicity, let's consider the mean function to be zero-mean, although it can be any arbitrary function[Ras03; Bar12].

The GP is then defined as follows:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x'))$$

Where:

- $f(x)$ is a function drawn from the GP.
- $m(x)$ is the mean function, often assumed to be zero for simplicity.
- $k(x, x')$ is the covariance function, also known as the kernel function, which determines the similarity between input points x and x'. Common kernel functions include the Radial Basis Function (RBF) kernel, linear kernel, polynomial kernel, etc.

There are various methods to measure or approximate uncertainty, such as Deep k-Nearest Neighbors [PM18], which is discussed in other sections. In this section, the uncertainty-based approach will focus on density and sampling-based methods.

Single uncertainty

The nearest neighbor algorithm has good explainability for its prediction. However, scalability remains a significant challenge especially when it comes to large datasets. The uncertainty-based approach can effectively manage large datasets by clustering data points based on their density, thus improving efficiency.

GPs with RBF kernels are renowned for their ability to capture uncertainties without being over-confident in their prediction. However, GPs face scalability issues, struggling with datasets beyond a few thousand points. Furthermore, compared to DNNs, GPs' kernels may not represent data as effectively.

Addressing these limitations, Bradshaw et al. [BMG17] introduced a Gaussian Process Deep Neural Network (GPDNN) model, combining the strengths of DNNs with GPs. This hybrid model leverages a GP layered over a DNN, integrating the predictive capabilities of both systems. Specifically, it uses a GP on top of the DNN model to combine the capability of both models.

Similar to Bradshaw's et al. [BMG17] idea, Sensoy et al. [SKK18] and Malinin el at. [MG18] applied a Dirichlet distribution over the class probability. This approach transforms classification tasks into predictions about distributions over possible softmax outputs, offering a richer interpretation than point estimate of a softmax output. It outputs a probability vector that indicates how likely it belongs to a certain class. Similar to softmax evaluation, the maximal value of the vector is used to represent the confidence score.

Van et al. [VST$^+$20] introduced an RBF network combined with a DNN, named Deep uncertainty quantification (DUQ), designed to minimize intra-class distances while maximizing inter-class separations. Uncertainty is quantified through the proximity of model outputs to the nearest class centroid. Liu et al. [LLP$^+$20] extended this idea and proposed a Spectral-normalized Neural Gaussian Process (SNGP), emphasizing distance-aware features within models. By applying spectral normalization across layers and integrating a GP with an RBF kernel at the final layer, they enhanced the model's sensitivity to distance, improving uncertainty measurement.

Van et al. [vASJ$^+$21] further developed these concepts and proposed addressing the issue of feature collapse identified in earlier models. By incorporating residual connections and spectral normalization, the model can better distinguish in and out-of-distribution data in feature space.

Aleatoric and Epistemic uncertainty

Distinguishing between different types of uncertainty is crucial for better understanding model performance and identifying areas for improvement. It also provides valuable insights to businesses on model behavior.

The Bayesian approach, one of the earliest methods for quantifying uncertainty, was improved by Kendall et al. [KG17], who proposed a Bayesian deep learning framework capable of modeling aleatoric and epistemic uncertainties separately. This approach introduces a variance variable to each output, trained via a maximum-likelihood loss (aka heteroscedastic loss), and employs Monte-Carlo sampling to predict both model and data uncertainties. However, this method requires architectural modifications and the use of heteroscedastic

Table 2.4: Summary of Aleatoric and Epistemic uncertainty related approaches

Paper	Year	Aleatoric uncertainty	Epistemic uncertainty
[KG17]	2017	Model output log variance	Monte Carlo
[TL19]	2019	SQR(Simultaneous Qunatile Regression)	OCs (Orthonormal Certificates)
[LSS20]	2020	Assumed density filtering	Monte-Carlo drop out
[CZG20]	2020	Normalizing Flow	Dirichlet distribution
[MKvA+23]	2023	Softmax entropy	Gaussian Discriminant Analysis

loss, which may not be feasible in all scenarios.

To capture aleatoric uncertainty, Tagasovska et al. [TL19] proposed the Simultaneous Quantile Regression (SQR) technique to estimate aleatoric uncertainty. This method employs a loss function to learn all the conditional quantiles of a target variable, enabling the computation of well-calibrated prediction interval. The aleatoric uncertainty is then estimated by the prediction interval around the median, calculated as the difference between the quantiles at $1 - \alpha/2$ and $\alpha/2$, where α is the significance level. To quantify epistemic uncertainty, the introduction of Orthonormal Certificates (OCs) was suggested: a collection of diverse non-constant functions that map all training samples to zero and signal epistemic uncertainty by mapping out-of-distribution examples to non-zero values. These certificates are trained to minimize a loss function that drives the dataset representations towards zero, with a regularization term added to enforce orthonormality among the certificates. Epistemic uncertainty is quantified by evaluating the mean square of the certificates' outputs.

Loquercio et al. [LSS20] proposed a general framework to predict both aleatoric uncertainty and epistemic uncertainty. The framework is based on Bayesian belief networks and Monte-Carlo sampling, it modifies the forward pass of a neural network to generate not only output predictions, but also their respective data uncertainties. As for the model uncertainty, it uses the MC approach with multiple passes, and computes the variance for the prediction. The predictive uncertainty is the combination of both uncertainties.

One of the potential issues mentioned earlier [SKK18; MG18] is that it requires OOD samples which in general is not always available. Furthermore, these methods assume arbitrary data distributions. Charpentier et al. [CZG20] propose a Posterior Network (PostNet), which assigns high epistemic uncertainty to out-of-distribution samples, low overall uncertainty to regions nearby observed data of a single class, and high aleatoric and low epistemic uncertainty to regions near observed data of different classes. PostNet integrates three key components: an encoder for latent space positioning, a normalizing flow

Table 2.5: Summary of ranking based approaches

Paper	Year	Target type	Target
Failure detection [HDV18]	2018	fixed	Binary [1,0]
Learning Confidence for out of distribution [DT18]	2018	flexible	N/A
True class probability [CTB+19]	2019	fixed	y_{true}
Confidence range [CBT22]	2022	flexible	[0,y_{true}] / [y_{true},1]
Correctness Ranking loss [MKS+20]	2020	flexible	Accuracy during training
DBLE [XAZ+19]	2020	flexible	Distance in the representation space
REL-U [GRP+23]	2023	flexible	Density
HUQ [VKT+23]	2023	flexible	Uncertainty

for density estimation in this space, and a Bayesian loss for uncertainty-aware training.

Continuing the work of [LLP+20] and [vASJ+21], Mukhoti et al. [MKvA+23] addressed the challenges of SNGP and DUE, notably the significant changes required in the modeling process and their inability to disentangle aleatoric from epistemic uncertainties. This method reduces architectural modification efforts by fitting a feature-space density estimator post-training, assessing epistemic uncertainty. For samples with low epistemic uncertainty, softmax entropy is used to evaluate aleatoric uncertainty.

2.2.4 Sample Ranking

The objective of those approaches is to accurately separate the misclassified and correctly classified samples irrespective of the source of error. In the context of selective classification, the main aim is to maximize coverage (or recall) while maintaining a predefined level of precision. So the objective in those approaches is to make sure correctly classified samples have a higher score than incorrectly classified samples.

One of the intuitive objectives is whether the model can classify the sample correctly or not. Hecker et al. [HDV18] and Blatz et al. [BFF+04] introduce a mechanism for predicting a correctness/failure score (0 or 1) for each sample in addition to the standard output. This additional target enables the model to learn from its predictions, distinguishing between correctly and incorrectly classified samples.

The limitation of previous [HDV18; BFF+04] work is that it does not differentiate difficult predicted samples from easily predicted samples, potentially hindering the model's ability to generalize. To address this, Corbiere et al. [CTB+19] proposed a new learning target which is called True class probability (TCP) where they designed a second model to predict the TCP from the classifier prediction instead of the failure of the classifier. In theory, the correctly classified samples will tend to have a high probability while the incorrectly classified sample will have a low probability.

Xing et al. [XAZ$^+$19] share a similar perspective with Corbiere's idea in [CTB$^+$19], employing a separate confidence predictor. It uses the distance to representation space generated from protocol learning as the confidence target, and it trains the classification model and confidence predictor simultaneously.

Devries et al. [DT18] designed a new architecture where the model can predict confidence score and probability at the same time. It utilizes a novel loss where the softmax prediction probabilities are adjusted by interpolating between the original predictions and the target probability distribution, where the degree of interpolation is indicated by the network's confidence.

Emerging methodologies focus on error-aware approaches that learn from training mistakes, adjusting confidence levels accordingly. Compared with Devries's [DT18] approach where it gives a hint to model to learn the confidence, Moon et al. [MKS$^+$20] introduced a confidence target that reflects accuracy, utilizing a ranking criterion where the probability of correctness is proportionate to the frequency of correct predictions during SGD-based optimization. This ensures samples that are easier to classify are assigned higher confidence targets than more challenging ones.

Gomes et al. [GRP$^+$23] propose an alternative method that leverages both positively (correctly classified) and negatively (incorrectly classified) instances to uncover patterns in the distribution of soft predictions. This enables the identification of misclassified samples through predicted class probabilities.

Vazhentsev et al. [VKT$^+$23] did not propose any learnable target. However, they combined the existing uncertainty approach and pre-existing methods for aleatoric and epistemic uncertainty, generating a total uncertainty score to represent its rejection score.

2.2.5 Ensemble approach

The Bayesian framework offers a practical tool to reason about uncertainty in deep learning [Nea12]. Nonetheless, training the Bayesian NN is a difficult task. Common approximation approaches such as variational inference [GG15; MVH$^+$21], Markov chain Monte Carlo(MCMC) [ZLZ$^+$19; ZA20] to the model posterior are often used instead.

However, those approximations in general are costly, and often deemed impractical for industrial applications. This has led to the exploration of ensemble methods as a simpler, yet effective, alternative for evaluating confidence scores.

Dropout [SHK$^+$14] can be viewed as an ensemble technique. Gal et al. [GG16] introduced the Monte Carlo dropout (MC dropout) as a means to

Table 2.6: Summary of ensemble related approaches

Paper	Year	Same model
MC Drop-off [GG16]	2016	Yes
Deep ensemble	2017	NO
Hyper deep ensemble [WST+20]	2020	NO
Anchor ensemble [PLB20]	2020	NO
Ensemble search [ZZE+21]	2021	NO
Repulsive deep ensembles [DF21]	2021	NO
Cascaded deep ensembles [XB23]	2023	N/A

approximate uncertainty by averaging the outcomes of multiple stochastic forward passes.

MC dropout essentially represents an ensemble of neural networks, where predictions are aggregated across several network instances. In contrast, Lakshminarayanan et al. [LPB17] suggested a straightforward deep ensemble method, focusing on randomizing various models before combining their outputs. Enhancing diversity among ensemble members, as suggested by Lee et al. [LPC+15], can significantly boost performance.

While deep ensembles primarily aggregate over model weights, Wenzel et al. [WST+20] introduced the concept of hyper-deep ensembles, which additionally incorporate hyperparameter variation to further enhance performance. Pearce et al. [PLB20] addressed criticisms regarding the non-Bayesian nature of deep ensembles by introducing the anchor ensemble approach, which regularizes parameters based on values from an anchor distribution.

Subsequent research by Fort et al. [FHL19] and Ovadia et al. [OFR+19] confirmed the effectiveness of deep ensembles in enhancing accuracy, uncertainty quantification, and out-of-distribution detection. However, the risk of insufficient diversity leading to model saturation remains a concern. Angelo et al. [DF21] introduced a kernelized repulsive term in the update rule of the deep ensembles to make sure the model discourages the sub-models from collapsing to the same function.

Zaidi et al. [ZZE+21] explored diversifying ensemble models through Neural Ensemble Search, aiming to identify a set of complementary architectures. Xia et al. [XB23] proposed a cascaded deep ensemble that each model will check against the threshold to decide whether to accept or reject. If the sample gets rejected, then it will pass to the next model. By doing so, it can balance the computational and model performance.

Despite these advancements, skepticism remains regarding the efficacy of deep ensembles, as voiced in Abe et al. [ABP$^+$22] and Theisen et al. [TKY$^+$24], encourage further research into the conditions under which ensembles yield optimal results.

2.3 Summary

In this chapter, we presented the definition of selective classification and its related work from the past five years. Furthermore, we categorized the related work based on its methodology, including: the softmax baseline and its regularization approach, the nearest neighbor approach, the uncertainty-based approach, the ranking-based approach, and the ensemble-based approach. Each approach has its advantages and disadvantages, and there is no consensus that one methodology outperforms the others in terms of performance. In the following chapters, we will discuss various problems and how the approach described here can be applied.

Chapter 3

Neural Machine Translation for Harmonized System Codes Prediction

The harmonized system codes (HS codes) are used world wide to categorize products in international shipments. In its basic form HS codes come in 6 digit format, subdivided hierarchically in groups of two digits (chapters, headings and subheadings). When shipping products, it is mandatory to specify a HS code for the purpose of producing a custom declaration. Currently the process is mostly carried out by human experts who take a decision on the HS code to be assigned to a shipment depending on the item description provided by the shipper. As such the process is time consuming and prone to errors due to generic, incomplete or non interpretable descriptions. The objective of this research is to automate the classification of HS codes in order to increase productivity to cope with extra volume in the customs classification area. For the purpose of testing the developed models, we used an anonymized data set of shipments provided by DHL. The main contribution of this paper is we tried a deep learning model that has not been tried to tackle the HS code classification problem: An attention-based neural machine translation(NMT) model with integration of the idea of hierarchical loss. The model can classify around 29% percentage of the dataset where the model's accuracy can reach 85%.

This chapter is based on the following publication:

Xi Chen, Stefano Bromuri, and Marko van Eekelen. *2021*, DOI: 10.1145/3468891.3468915, Neural Machine Translation for Harmonized System Codes prediction. In Proceedings of the 2021 6th International Conference on Machine Learning Technologies, ICMLT.

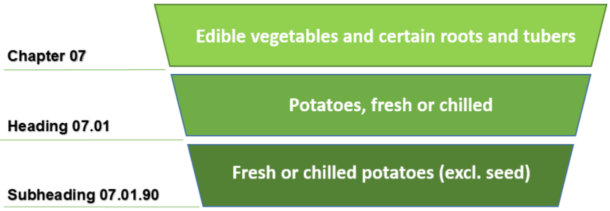

Chapter 07
Edible vegetables and certain roots and tubers
Heading 07.01
Potatoes, fresh or chilled
Subheading 07.01.90
Fresh or chilled potatoes (excl. seed)

Figure 3.1: HS Code illustration

3.1 Introduction

The harmonized system codes (HS codes) are used worldwide to categorize products. Figure 3.1 illustrates the structure of an HS code, which is composed of six digits. In its basic form, HS codes come in a 6-digit format (i.e. 07.01.90), subdivided hierarchically into groups of two digits (chapters 07, headings 07.01, and subheadings 07.01.90). When importing or exporting goods, it is mandatory to provide its associated HS codes to customs clearance. Currently, the majority of the work for assigning HS codes is done by domain experts. It is a labor-intensive and error-prone task. Thus, automatic classifying of HS codes is needed in order to provide an aid to domain experts by proposing/predicting the HS Code.

In the logistics sector shipments have to be described by a number of attributes, such as origin, destination, shipper and item description of the items being shipped. As such, our hypothesis is that the problem of producing an HS code classification starting from item description and features of the shipment can be considered as a machine translation problem [SVL14], where the HS code space represents the target language to be produced starting from the features and description of the line item being shipped.

In addition, the codes take a hierarchical structure, with dependencies occurring between the chapters, headings and sub headings, therefore another possible modeling could imply a hierarchical classification model [SL01; SF11].

Thanks to deep learning models such as the neural machine translator [SVL14] (NMT), machine translation has seen a quick advancement in recent years. NMT models rely on recurrent neural networks (RNN) often in the form of long short term memories [HS97], and are usually modelled with an encoder-decoder architecture, where the encoder is fed with sentences of one origin language, plus additional features, and the decoder is fed with sentences of the target language. Further advancements have been achieved thanks to the conception of attention models. Attention models allow to focus the network on parts of subparts of the sequence and have shown to greatly improve tasks such as image captioning and language translation [VSP+17].

The problem of classifying HS codes has been previously recognized as a research problem [WYY+06], but it has been tackled using exact word matching and ontologies, with limited ability to generalize to unseen descriptions. The main contribution of this paper with respect to the state of the art is to model the problem of classifying HS codes as a machine translation task, where the input language comprises the description of the shipment, its origin and destination and the output language is the HS code associated with the shipment. In addition to this, we also modeled a hierarchical loss specific for the HS code task that allows us to improve over the basic NMT model. This is significant because the proposed model and hierarchical loss allow classifying automatically around 29% of the data used for the experimentation with an accuracy of 85% for codes comprising 6 digits, where the human expert accuracy has been estimated to be between 65% and 75%.

The rest of the paper is structured as follows. Section 3.2 discusses relevant related work. Section 3.3 discusses the data sample used and Section 3.4 discusses the method applied in this contribution. Section 3.5 discusses the results concerning and the limitations of the study. Section 3.6 concludes this paper proposing potential future work directions.

3.2 Related Work

Most of the work about HS codes concerns the definition of a knowledge base for manual search of the code, given a shipment description, for example Wei et al. [WYY+06] use an ontology based service to help a user generating the right code given a product. Singh [SS04] adopted the fuzzy logic to help identify wrongly classified HS codes. Ding et al. [DFC15] follow a fuzzy logic approach by applying a background net (a model that dynamically builds a network of words and their co-occurrences) to the automatic classification of HS codes, showing that a statistical approach can lead to better results than exact keywords matching.

Concerning classification, the HS code classification problem can be formulated as a hierarchical classification problem. There are three different approaches for hierarchical classification: flat approach, hierarchical local approach [KS97; SL01] and hierarchical global approach [SF09; KMF+05]. The flat approach addresses the hierarchical problem as a multi-class classification. The global local approach, a top-down structure like a tree is specified in which each node requires a local classifier. Instead, the hierarchical global approach is utilizing one model and try to classify all at once. Figure 3.2 illustrates the difference of those three approaches.

The main problem of flat classifiers is that they ignore the hierarchical structure. The local approach [CH04; CGZ06] only takes partial hierarchical

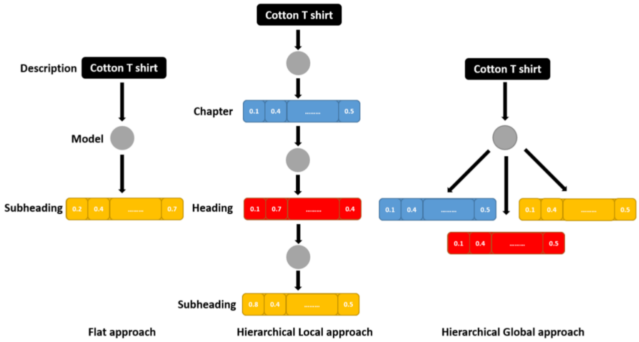


Figure 3.2: Hierarchical Model

information into account since all the local classifiers are isolated between each other. Recently, the global neural network based approach is prevailing when dealing with hierarchical classification tasks. This approach simulates the hierarchical structure by using neural network and modifying the loss function accordingly [WS17; Gao20; WCB18; KBH+17; MTH+19] in order to make sure the model will capture the information on global level.

Another field of research closely related with this problem is the one of multi-label learning [Cha20]. Compared to standard classification approaches, in multi-label learning the items can have multiple labels at the same time. Several approaches exist to model the presence of multiple labels, such as for example binary relevance [ZLL+18], chain classifiers [JLL+19] and also multi-label deep learning architectures [NMK+17; US19]. Modeling the HS code automatic classification with a multi-label approach would imply defining an encoding for the chapter, heading, subheading sections of the HS code, that is not very practical due to the fact that each of these sections can in principle have many sub-labels (up to 100), with in addition the problem of considering dependencies between these labels, following an approach similar to the NMT. As a matter of fact, previous versions of our architecture used a three-output neural network based on LSTMs to classify the HS code, but we found no improvements with respect to hierarchical classification or the NMT approach presented in this paper.

The NMT structure that is being applied in this research itself has the advantage of carrying the hierarchical information while maintaining label consistency.

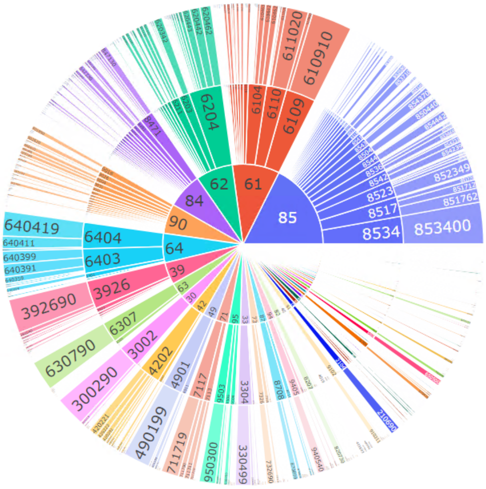


Figure 3.3: HS code distribution

3.3 Data

The data used in this experiment develops in a period of eight months of shipments towards one country via the DHL network. The data set contains the following features: item description, origin, destination, origin airport, destination airport. It has 1.156 million records. Among those records, there are 476128 (41.18 %) unique descriptions, 705 origin airports, 40 destination airports, 183 export countries, 13014 different combination of origin airports and destination airports, 4257 different HS code in six digits level and the distribution of it shows in Figure 3.3. Since this work was prepared during the COVID-19 outbreak, due to the influence of the pandemic, the description that contained masks, kn95, and blood samples has been removed in order to make the conclusion more generalizable. The infrequent HS codes which appear less than 10 times also got removed in the cleaning step. Regarding the preprocessing of the text descriptions, we applied a standard NLP approach involving the conversion of every description and text field into lowercase, and removed the punctuation and the digits.

It is worth to mention that the data set is not clean. There are two main issues: first of all, often the description does not contain enough information to classify six-digit HS code. Secondly, part of the HS codes are assigned wrongly due to human mistakes. In order to address those two issues a parallel work to this one is focusing on implementing a description quality measurement, but the details of this development are beyond the scope of this paper.

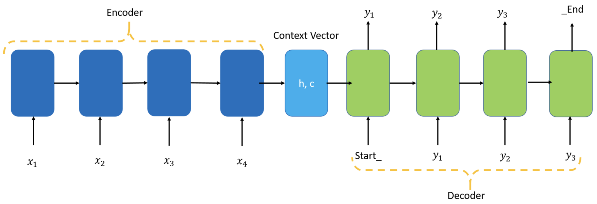


Figure 3.4: NMT Architecture

3.4 Methods

Three different architectures are being applied in this experiment: Hierarchical logistic regression, LSTM and NMT. We briefly describe these three approaches in this Section.

3.4.1 Logistic Regression

The first approach that is being applied is Hierarchical Multinomial Logistic Regression(HLR) [Böh92]. It is a local hierarchical approach where we build a multinomial logistic regression model at each node as showed in Figure 3.2.

For each multinomial logistic regression model, we expand on the node where it has the largest probability:

$$P(Y = K) = \frac{e^{(\beta_i * X_i)}}{\sum_{k=1}^{K} e^{\beta_k * X_i}} \tag{3.1}$$

K is the total possible output, β is the coefficient[Böh92].

3.4.2 Neural Machine Translation

A neural machine translator (NMT) is an encoder-decoder model for sequences, and it is meant to translate one sequence (i.e. an English sentence) to another sequence (i.e. the respective French sentence). Formally, the NMT transforms sequences of vectors $x = (x_1, x_2, \dots, x_{Tx})$ into sequences of vectors $y = (y_1, y_2, \dots, y_{Ty})$ where the sequences may not necessarily have the same length (i.e. T_x different from T_y). The translation is often performed using an RNN network (for example an LSTM), due to their ability of representing sequences. Other architectures are possible, for example a combination of RNN and CNN can also be used. In the case of the RNN, in the decoder part of the NMT architecture, the the hidden state of the RNN at time t, h_t is used to define a context vector $c = q(f(h_1, \dots, h_{Tx}))$, where f and q are non-linear functions. The context vector effectively encodes all the information of the sequence. The training of the decoder part of the network, that is usually also

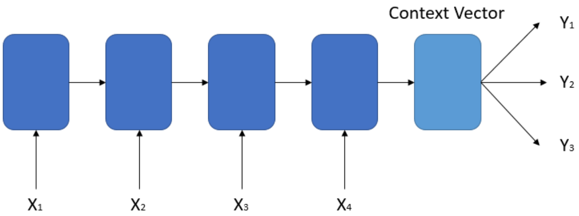


Figure 3.5: LSTM model structure

represented by means of an RNN, happens by predicting the next word $y_{t'}$ given the context vector c and all the previously predicted words $y_1, y_2, \ldots, y_{t'-1}$.

Formally this can be written as:

$$p(y) = \prod_{t=1}^{T} p(y_t | y_1, y_2, \ldots, y_{t'-1}, c)$$

In terms of an RNN, this probability is expressed as

$$p(y_t | y_1, y_2, \ldots, y_{t'-1}, c) = g(y_t - 1, s_t, c)$$

where g is a nonlinear, potentially multi-layered, function that outputs the probability of y_t, and s_t is the hidden state of the RNN.

The context vector c of the NMT has the disadvantage of behaving like a bottleneck concerning the information contained in the sequence. Effectively speaking the fixed length of the vector reduces the ability of the NMT to remember long sequences, often forgetting important parts of the sequence. Attention models have been introduced to reduce this effect. This paper makes use of additive attention [BCB15], that makes use of an explicit layer of neurons to reweigh the importance of certain parts of a sequence. Formally, the function of such a layer can be expressed as:

$$f_{att}\left(\mathbf{h}_i, \mathbf{s}_j\right) = v_a^T \tanh\left(\mathbf{W}_a\left[\mathbf{h}_i; \mathbf{s}_j\right]\right)$$

meaning that additive attention learns to align hidden states of the decoder (s_j) with hidden states of the encoder (h_i). In doing so, the attention layer also learns parameters v_a and $\mathbf{W}_a$. The final scores of the alignment are calculated by means of a softmax layer that reweighs the importance of the hidden states in the prediction.

3.4.3 Hierarchical loss

The NMT model itself trained in a teaching force [WZ89] way. It could be argued that it might not necessarily learn the global information considering

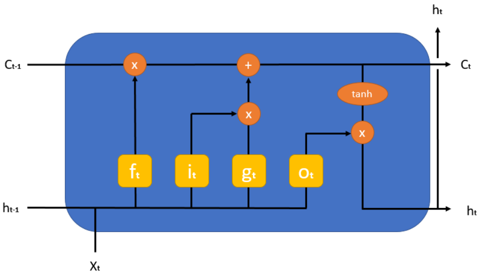


Figure 3.6: Long time short memory

it evaluates the model in the single output level instead of in the sequence level. So the idea of hierarchical loss [Gao20] is also being introduced in for addressing the hierarchical classification, which can be thought as a global loss function. The loss function is defined as follow:

$$L(y, \hat{y}) = \alpha * \sum_{k=0}^{l} y_k * log(\hat{y}_k) + \beta * H$$

The α and β are the hyper-parameters, y_k and $\hat{y}_k$ are true and predicted value at the k_{th} digits respectively. H is the binary value, it equals to 0 for all k if $y_k = \hat{y}_k$, 1 else.

3.4.4 Long Short-Term Memory (LSTM)

In order to identify the gain from the NMT decoder structure. We also tested an LSTM model which has the same structure as the NMT encoder. The structure of the model is showed in Figure 3.5. The LSTM cell will take one input at a time, then it will predict the chapter, heading, subheading at one goal after it went through all the input.

The LSTM is one type of RNN, and it was introduced in order to solve the vanishing gradient issue that vanilla RNN suffered from. The LSTM cell contains three different gates to control the information flow. They are: input gate i_t, forget gate f_t and output gate o_t respectively as Figure 3.6 shows. Each gate state is determined by its internal states: weights W and bias b , output from the previous state h_{t-1}, and the current input X_t.

The way it updates the state:

$$\begin{aligned} i_t &= sigmoid(W_{ix} \cdot x_t + W_{ih} \cdot h_{t-1} + b_i) \\ f_t &= sigmoid(W_{fx} \cdot x_t + W_{fh} \cdot h_{t-1} + b_f) \\ o_t &= sigmoid(W_{ox} \cdot x_t + W_{oh} \cdot h_{t-1} + b_o) \end{aligned} \tag{3.2}$$

The internal state will also get updated during each step:

$$C_t = f'_t \times C_{t-1} + i_t \times tanh(W_c \cdot x_t + b_c) \tag{3.3}$$

The output at each step is:

$$h_t = o_t \cdot tanh(c_t) \tag{3.4}$$

3.5 Results

In this experiment, we evaluated five different models: HLR, LSTM , LSTM with hierarchical loss(LSTM-HL), NMT and NMT with hierarchical loss(NMT-HL). The results are compared in two dimension: the percentage of the data that can be auto-classified and the accuracy on that scope.

The data $X = \{x_1, x_2, ..., x_n\}$, is analyzed with respect to a confidence score $P = \{p_1, p_2, ..., p_n\}$ where p_n is the model's predicted probability for the data point x_n. A threshold T is applied on confidence score p_n to obtain a certain desired accuracy A. X'_{val} is the auto-classify percentage, a subset of validation data set X_{val}, where $X'_{val} \in X_{val}$ and its $P'_{val} >= T$

The threshold is calculated on the validation dataset and result is evaluated on the test dataset by applying the same threshold.[JKG+18] In order to make the result closer to reality, we split the data temporally, using all the data, except for the last two months, for training, and using the last two month of the data for validation and testing respectively.

	HLR		NMT		NMT-HL		LSTM		LSTM-HL	
Accuracy	Acc	Recall	Acc	Recall	Acc	Recall	Acc	Recall	Acc	Recall
N.A	**45.95**	100	45.00	100	45.42	100	43.08	100	41.94	100
70	69.40	52.95	**70.44**	53.53	70.24	**55.50**	70.72	54.05	69.76	53.04
75	74.54	44.12	75.52	45.44	**75.65**	**47.26**	75.71	46.30	74.56	44.69
80	79.69	35.43	80.62	37.61	80.70	**39.15**	80.45	38.74	**80.76**	36.83
85	85.15	27.09	85.41	29.83	**86.09**	29.52	85.03	**30.38**	85.17	29.58
90	91.06	18.40	90.57	**21.20**	**91.35**	20.39	90.95	20.96	91.13	21.13

Table 3.1: Model comparison on accuracy and recall. The first columns is the desired accuracy, it is used for calculating the threshold for the confidence score on the validation dataset. The accuracy for each model is the accuracy where its probability is larger than the threshold.

3.5.1 Result analysis

The result is showed in Table 3.1. If no limitation (N.A) is imposed on accuracy, HLR has the best performance among all models. The NMT-HL performs the best recall when the wanted accuracy is at 70%, 75% and 80%. The deep learning models have more or less the same accuracy and recall at the accuracy threshold of 85% and 90%. In general, the deep learning models have better performance compared to the HLR model at the accuracy threshold above 70%.

The additional hierarchical loss is improving the NMT model's performance, but it deteriorates the result of the LSTM model. One explanation could be without hierarchical structure or direct connection in the predicted chapter, heading and subheading, the additional penalty might just confused the model.

The overall difference on those five models are not big, this could come from the following reasons: first, the data is noisy (i.e. wrong chapters, headings and subheadings), and this might have impact on the models; secondly, the majority of the descriptions are short and do not have language structure, limiting the effectiveness of RNN networks. Also, the dataset is not that large considering it has 4257 different HS code combinations, the deep learning model might benefit more if more data can be provided.

3.6 Conclusion

In this experiment, we tested five models: HLR, LSTM, LSTM-HL, NMT and NMT-HL model for predicting the HS Codes description, and established the possibility of applying selective classification in this problem. Currently, all the HS Codes classifications were done by the agent manually. Based on the results, we can automate 29.52% of the descriptions in the current data set with a precision of 85%. This is significant considering the millions of shipment descriptions that need to be classified monthly in a large logistic company like DHL. It can save lots of effort, improving the agent's productivity and accuracy.

In terms of future work, several questions can be further investigated in the future. From the standpoint of embeddings, we can fine-tune the pre-trained embedding on the current data instead of training it from scratch, so that we can utilize more advanced embedding models like BERT [KT19] and XLNET [YDY+19]. A transformer network [VSP+17] has been already attempted by the authors, but no improvement compared to HLR could be found, albeit the detailed analysis and fine-tuning still need to take place. A potential model to try is a Transformer-based pre-trained model [DYW+19; STQ+19]. Also, the positional encoding [VSP+17; GAG+17] that is being applied on most of the latest seq2seq models [SUV18; TO19] might have some impact if added to the decoding part on NMT model. Additionally, we could also investigate further the probability of the model and see the alternative in determining whether or not we should trust the model's prediction [JKG+18; ASS+20].

Chapter 4

Pair-wise selective classification for shipment importer prediction

Whenever a shipped package crosses the border, logistic companies have to declare importer information for the clearance process. This information is not always provided by the customer, causing delays and additional expenses. Fortunately, importer information can often be inferred from historical shipments. The current technical standard, even in big companies, is to use a feature-weighted nearest neighbor approach based on domain knowledge. Nearest neighbors assume that each sample point is represented independently and is fixed in some high dimensional space. This makes it difficult to integrate higher-order pair-wise relationships such as transaction frequency from shipper to receiver transaction, because the features are now changing dependent on the pairs. This would require a complex ad-hoc feature engineering and metric learning to capture pairwise relationships properly with nearest neighbors. In this paper, we propose a framework for importer prediction based on a pair-wise classification approach that allows us to capture higher order pair-wise relationships. We also incorporate an auxiliary neural network that can reliably reject shipments that our model could not predict well such as shipments with new importers that are not in the historical data. This allows us to pass the difficult cases to a human agent instead of naively making an incorrect prediction. Our proposed pair-wise solution outperforms the industry standard by a significant margin of precision across a wide range of recall values.

This chapter is based on the following publication:

Xi Chen, Daniel Stanley Tan, Prakash Gupta, and Stefano Bromuri. *2023*, DOI: 10.1145/3587716.3587741, Pair-wise selective classification with dynamic sampling for shipment importer prediction. In Proceedings of the 2023 15th International Conference on Machine Learning and Computing, ICMLC.

4.1 Introduction

International packages shipped via a logistic company are required to declare the importer information as part of the customs clearance process. However, this information is not always provided by the customer. Fortunately, for repeating shippers and importers, we can derive this importer information based on their transaction histories.

From a machine learning perspective, we can formulate this problem as identifying and retrieving the most similar shipments. There are different distance functions that can be applied in here such as Euclidean distance and feature weighted distance [VHdC$^+$07; CH17]. Currently, logistic companies use a feature-weighted nearest neighbour (FWNN) approach [VHdC$^+$07; KLY97], to determine the importer identity, and reject the prediction if the similarity is less than a certain threshold in order to meet the precision requirement defined by the business. However, the issue of FWNN models is that they can not cope with complex combinations of multiple rules or higher-order relationships between different shipments, leading to sub-optimal performance when predicting the importers.

Given this limitation, we propose a pairwise binary classification task that takes two shipments' information and statistics as input to predict whether or not they come from the same importer. This allows us to incorporate higher-order pairwise interactions between shipments. The main challenge to train such a classifier is to construct a training data set with appropriate positive and negative training pairs, considering that there are millions of negative samples for each shipment record. This requires to apply sampling strategies, the details will be explained in Section 4.2.1, but we use a dynamic sampling strategy [ZCW$^+$13] by sampling positive and negative pairs based on a score function.

After the classifier is trained, a rejection approach for shipments with uncertain or new importers is needed in order to meet the accuracy requirement from the business and legal parties. For FWNN, a threshold is implemented based on a similarity distance. As for our proposed pair-wise classifier, the baseline is a threshold on the predicted probability. There are various strategies of selective classification [GE17] that can be adopted, which will be covered in Section 4.2.2. We use an auxiliary confidence network trained with True Class Probability (TCP) [CTB$^+$19] to predict the confidence score for rejection purpose in this paper.

From an industry perspective, time efficiency is also important. So in order to reduce the inference time, we also introduce an exact matching logic to predict the easy samples before the classifier.

In summary, our contributions are:

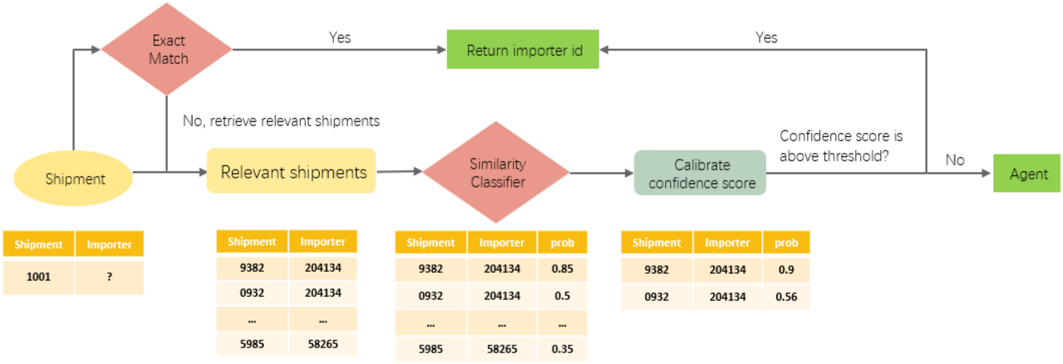


Figure 4.1: Pipeline for importer prediction.

1. We propose a pair-wise binary classification schema for determining the importer, which takes pair-wise relationship into consideration.
2. We incorporate a confidence network on top of the classifier, which allows for selectively classifying samples that can reduce costly mistakes.
3. The results show that our proposed model outperforms the feature-weighted nearest-neighbor industry standard by a significant margin of precision across a wide range of recall values, and multiple algorithms.

The rest of this paper is structured as follows: Section 4.2 discusses relevant related work; Section 4.3 discusses the main technical methods used in this paper; Section 4.4 discusses the data, and evaluation of the proposed approaches against the baseline; Section 4.5 concludes this contribution by summarizing its relevance and pointing out potential future work.

4.2 Related work

In the problem presented in this paper, when constructing the training pairs to train the model, each sample may have several thousand positive samples (shipments that have the same importer) and millions of negative samples. So an effective sampling strategy is needed in order to select the proper negative and positive samples.

Sampling approaches are used in many different domains to deal with complex data-related issues [KB19; DQY$^+$20], but potentially the closest field to the importer prediction problem is that of recommendation systems [DQY$^+$20; ZZH$^+$22; DQH$^+$19], in the case in which only implicit feedback is observed, and the target is to learn a personalized ranking. This is similar to our case as both of the problems are facing the same challenge, where the population of negative samples is really large, thus an effective sampling is needed.

Other close related work concerns the rejection option of the classifier (also known as selective classification) [Cho70; BW08; JKG^{+}18]. As the importer identity is legally bounded to the shipment, it is important to make sure the classifier's performance can reach a predefined level imposed by business requirements. Several rejection approaches will be explained later.

4.2.1 Negative sampling

Recommendation is an important task in many applications. From shopping websites to Vlogs. Implicit feedback such as clicks is quite common. There are many proposed methods that address this problem by means of different technical means: one successful technique uses matrix factorization (MF) [SKK^{+}02], another successful technique uses adaptive k-nearest-neighbor (kNN) [PZC^{+}08; ZZH^{+}22]. However, these methods do not optimize the ranking directly. Rendle et al. [RFG^{+}09] proposed a bayesian personalized ranking(BPR) approach to directly optimize the personal ranking. Namely, the model learns the ranking relationship from the constructed positive and negative pairs. One of the issue that arises from BPR is that it samples uniformly the negative samples with respect to each positive sample, disregarding the problem of rank bias [ZCW^{+}13].

Directly extending on Rendle's work, there are two directions to apply a better negative sampling strategy: the first method implies using a heuristic sampling, employing a model to select the negative sample from the database [ZCW^{+}13; DQY^{+}20; ZZH^{+}22]; the second method is a model-based sampling where a deep generative model is trained to generate the sample [CKK^{+}18; PC19; DQH^{+}19]. The idea of these two techniques is to try to generate/identify the *hard negative* as this proved to be a more efficient sampling for model training [PC19].

The issue that arises from uniformed sampling is that it suffers from the vanishing gradient issue, because the gap between positive and negative samples is too large to provide valuable information for the training of a machine learning model, thus taking a long time to converge [ZZH^{+}22]. In order to identify the hard negative samples, [ZCW^{+}13; RF14] both proposed a dynamic sampling approach using a score function to determine the rank of the samples, and take this into account during the negative sampling. However, this type of sampling might end up selecting false negatives, which may lead to poor generalization. Subsequent research [DQY^{+}20; ZZH^{+}22] proposed solutions to overcome this problem. This paper focuses on dynamic sampling without taking false negatives into consideration approach, as we noticed these do not constitute an issue from an empirical perspective since the training data is checked by Customs.

4.2.2 Selective classification

Selective classification, also known as classification with rejection option, has been used in machine learning research for a long time [Cho70]. There are two types of rejection options. One is a cost-specific rejection option, where one assumes the cost for wrongly classified samples is known [CDT$^+$95; BW08]. The other, which is more common nowadays, is a threshold-based rejection option. This utilizes a confidence/trust/reliability score [CDT$^+$95; JKG$^+$18] to be compared with a selected threshold to decide concerning the rejection of the result. In this experiment, the threshold-based rejection option will be applied due to the fact that different countries might have different requirements concerning shipments. So the threshold-based solution gives the flexibility to adapt the technical solution to different business requirements.

K-nearest neighbors (KNN) methods [Ati05; Dal09] are a natural choice for producing confidence scores with good explainability associated with the prediction. Some variations have been made in order to adapt KNN selective classification with deep neural networks (DNNs), including combining KNN with DNNs. The approaches that have been tried include: using softmax output to represent the weights for the KNN model [Ati05], applying KNN on a hidden layer on the DNN [MAN17], and applying KNN on all hidden layers and aggregating the results [PM18; LCA$^+$20]. Similarly, Jiang et al. [JKG$^+$18] used the distance ratio between the nearest class to the predicted class, while Lee et al. [LCA$^+$20] replaced the last layer with a KNN classifier.

In addition to KNN-based techniques, another approach involves working with the softmax output directly. Vasconcelos et al. [VFB95] proposed a more reliable network by replacing the sigmoid activation function with a Gaussian activation function. Lately, Hendrycks et al. [HG16], and Geifman et al. [GE17] used the maximum value from softmax to represent model reliability and showed that the softmax output in DNN acts as a good baseline. Hecker et al. [HDV18] trained the model with additional output to predict whether the model would fail or not. Similar to Hecker, instead of predicting the failure of the prediction model, Corbiere et al. [CTB$^+$19] proposed an auxiliary confidence network to predict the samples TCP. Chen et al. [CBT22] proposed a confidence range target that bridges failure detection and TCP approach. In this paper, we choose TCP because of its high performance and simplicity. The best rejection approach for the importer problem could be defined with an ad hoc technique, but this is subject to future work.

4.3 Methodology

Figure 4.1 shows an overview of our framework. It consists of two components: Exact rule matching (Section 4.3.1) and similarity classifier (Section 4.3.2), with a confidence network (Section 4.3.5) for rejecting samples. The details of each component will be explained in the following section.

4.3.1 Exact Rule Matching

If new shipments perfectly match all the features or information of a previous shipment, then we can easily determine their importer. However, in many cases, only a subset of the features match previous shipments. The problem now becomes identifying the combinations of features that we can use for exact rule matching to reliably determine the importer details.

Since the combinations of features exponentially increase with the number of features considered, it quickly becomes infeasible to iterate through all possible combinations. Therefore, we use beam search to efficiently determine the matching rules, as shown in Figure 4.3.

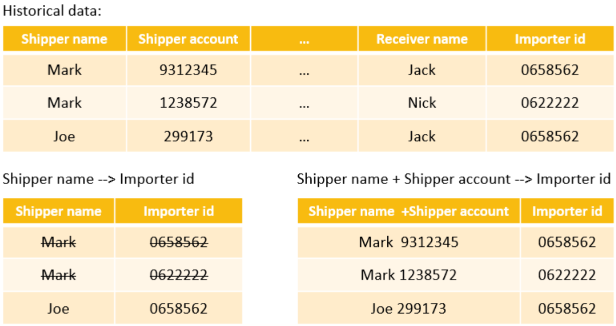
For each feature or feature combination, as long as the value or combination of the values is mapped with a unique importer in historical data, it will be added to a dictionary as key and value pair respectively as shown in Figure 4.2.

To identify the rules, we evaluate all the single features and compute their precision and recall performance in determining importers. Next, we expand only the top k to pairwise combinations of features, which significantly reduces the combinatorial explosion of features. We repeat this process for longer combinations of features until the longest possible combinations have been reached. The feature combination that has the best performance will be stored as the initial exact matching rule. Then, we remove the data that can be captured by the initial rule from the training data set, and iteratively search for new rules on the rest of the data set until the precision and recall can not meet the business requirement. This procedure leads to a sequence of exact matching rules.

These rules can be implemented with dictionaries or hash maps, which can determine the importers in a constant running time.

4.3.2 Pair-wise Classification

While the exact matching rules work well, there can be shipments that do not satisfy any of the exact matching rules. We propose a pair-wise similarity classifier (Section 4.3.3). Given any two shipments information, this classier predicts whether two shipments belong to the same importer or not. This



Historical data:

Shipper name	Shipper account	...	Receiver name	Importer id
Mark	9312345	...	Jack	0658562
Mark	1238572	...	Nick	0622222
Joe	299173	...	Jack	0658562

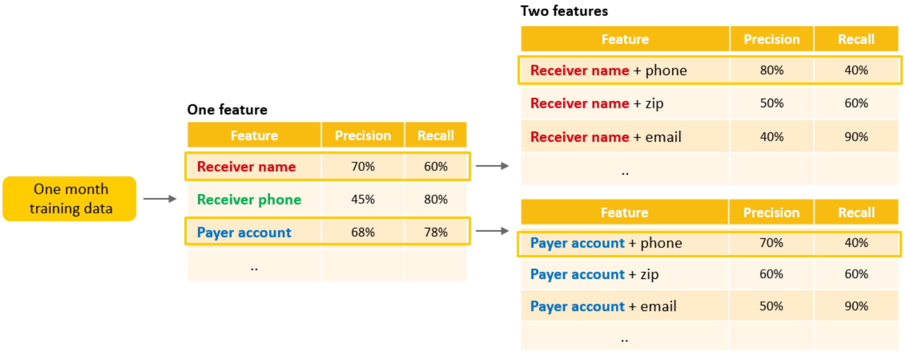
Shipper name --> Importer id

Shipper name	Importer id
~~Mark~~	~~0658562~~
~~Mark~~	~~0622222~~
Joe	0658562

Shipper name + Shipper account --> Importer id

Shipper name +Shipper account	Importer id
Mark 9312345	0658562
Mark 1238572	0622222
Joe 299173	0658562

Figure 4.2: Exact match mapping extraction



One feature

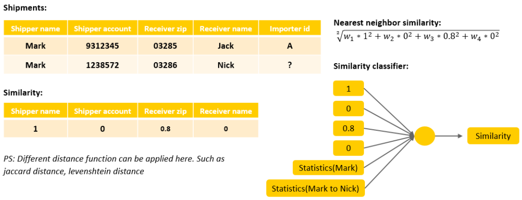
Feature	Precision	Recall
Receiver name	70%	60%
Receiver phone	45%	80%
Payer account	68%	78%
..		

Two features

Feature	Precision	Recall
Receiver name + phone	80%	40%
Receiver name + zip	50%	60%
Receiver name + email	40%	90%
..		

Feature	Precision	Recall
Payer account + phone	70%	40%
Payer account + zip	60%	60%
Payer account + email	50%	90%
..		

Figure 4.3: Beam search on rules.



Shipments:

Shipper name	Shipper account	Receiver zip	Receiver name	Importer id
Mark	9312345	03285	Jack	A
Mark	1238572	03286	Nick	?

Similarity:

Shipper name	Shipper account	Receiver zip	Receiver name
1	0	0.8	0

PS: Different distance function can be applied here. Such as jaccard distance, levenshtein distance

Nearest neighbor similarity:

$$\sqrt[2]{w_1 * 1^2 + w_2 * 0^2 + w_3 * 0.8^2 + w_4 * 0^2}$$

Similarity classifier:

Figure 4.4: Similarity feature.

classifier can easily integrate distances of different features, statistics of shippers, and the pair-wise relationship between shipper and receiver. We also incorporate a dynamic sampling strategy (Section 4.3.4) for sampling pairs that are most useful for training the model.

4.3.3 Pair-wise feature generation

With the pair-wise framework, we convert nominal features into a single value that denotes the similarity between the two shipments, as shown in Figure 4.4. This significantly simplifies the representation of nominal values since we do not need to represent each unique value. Additionally, we integrate statistics of the transaction history between shippers and receivers as part of the features for the classification. This is can not be easily done under a standard classification framework, since these features dynamically change depending on shipper and receiver.

Figure 4.4 illustrates the difference between the FWNN approach and the similarity classifier. The similarity classifier can integrate more hand-crafted features and it can utilize more complicated logic when computing the overall similarity score.

4.3.4 Dynamic Sampling

Our pair-wise classifier requires a pair of positive and negative samples for training.

However, for every positive pair, there are several orders of magnitudes more negative pairs, making the data heavily imbalanced. We need an efficient sampling strategy to ensure that the classifier trains properly. Inspired by Zhang et al. [ZCW+13], we introduce a dynamic sampling of both positive and negative pairs.

Figure 4.5 illustrates how we used dynamic sampling for the importer prediction task. Specifically, we first select positive and negative samples randomly. Next, we employ the score function $s(x)$ to score the samples. Specifically, we use the output probability of the currently trained model to

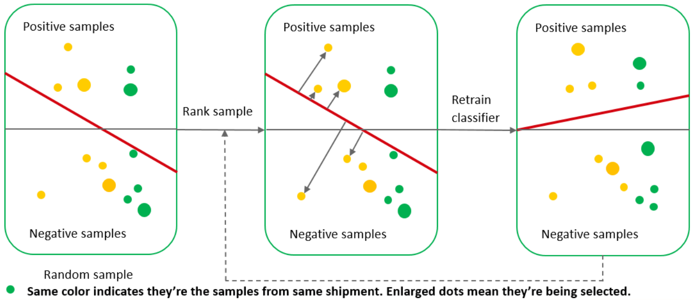


Figure 4.5: Dynamic sampling in the importer prediction task.

score each sample. Then, we re-sample based on the score. The lower score will have a higher probability of getting sampled. Sampling the difficult samples will help the model to converge faster and identify a better decision boundary as explained in Section 4.2.1. We show the more detailed steps in Algorithm 1.

Algorithm 1 Dynamic sampling for both positive and negative samples

```
Require: positive samples $(p_1, p_2, ..., p_i)$, negative samples $(n_1, n_2, ..., n_j)$,
    score function $s(x)$
    Query $s(p_1), s(p_2), ..., s(p_i)$
    Query $s(n_1), s(n_2), ..., s(n_j)$
    Return one positive sample from $(p_1, p_2, ..., p_i)$ with probability
    $(s(p_1), s(p_2), ..., s(p_i)) / \sum_{k=1}^{i} s_(p_k)$, one negative sample from $(n_1, n_2, ..., n_j)$
    with probability $(s(n_1), s(n_2), ..., s(n_j)) / \sum_{k=1}^{j} s(n_k)$
```

The whole process repeats until the model converges, i.e., the positive and negative sample probabilities do not change.

4.3.5 Confidence Network

To meet business requirements the classifier should reject samples that can not be predicted with high confidence, in order to let humans handle these shipments (e.g. new importers, or hard samples). In this way, the prediction performance on the processed data can still meet the precision standards required.

The easiest approach is to use the predicted probability to represent the sample's confidence score [HG16], and then apply a threshold to reject low-confidence samples. However, this does not consider the actual performance of the classifier and has been shown to make mistakes with a high confidence score associated [CTB+19].

We instead use the predicted True Class Probability (TCP) [CTB+19] to

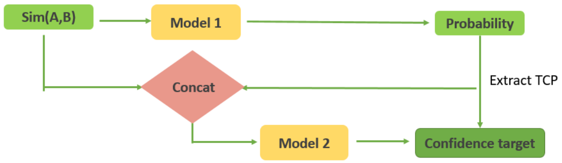


Figure 4.6: True class probability Training process. Model 1 is a probabilistic binary classifier, while model 2 will calibrate the predicted confidence score. After the model is trained, the sample will feed into both model 1 and model 2 sequentially, the final output is the predicted probability.

represent the sample's confidence score, with a confidence network classification schema as shown in Figure 4.6.

For a given input x, a standard approach is to compute the maximal probability of the predicted class $\hat{y}$ Probability: $P(Y = \hat{y}|w, x)$. The TCP target is, on the other hand, the probability of the ground truth label y^* probability: $P(Y = y^*|w, x)$. This works because the correctly classified sample's ground truth probability will tend to be higher, while the incorrectly classified one will be on the lower side, therefore the additional prediction model will work towards correcting this bias. In this contribution we discuss implementations of the confidence network using random forests and neural networks.

4.4 Experiment

4.4.1 Data

The data used in our experiments comes from one of the top logistics companies. We use ten months' worth of shipments toward one destination country as our case study. This is further split into training, validation, and testing sets. Specifically, we use 10 fold time series cross-validation to validate our approach in order to avoid information leakage. The rolling window size is half a month. The results that reported are the average of the 10 fold. Each month has around 150k shipments in average. Each shipment has the following features from both shipper and receiver: name, phone number, account, country, street, email, and zip code. The data set contains 296,000 different importer ids.

4.4.2 Setup

As illustrated in Figure 4.1 exact rule matching is first applied on the test set, to process obvious matches. We only evaluate the classifiers on the rest of the

Table 4.1: Exact rule matching results

Rule	Precision	Recall
Rule 1	99.53%	62.82%
Rule 2	98.30%	1.63%
Rule 3	96.09%	1.97%
Overall	99.40%	66.32%

Table 4.2: Recall under different precision requirements

Precision	FWNN	RF	RF + TCP	NN	NN + TCP
80%	7.17	16.95	22.96	22.13	**23.76**
85%	5.33	14.47	**20.65**	19.34	20.39
90%	3.94	10.47	15.45	14.34	**17.08**
95%	0	0.38	4.09	4.09	**5.76**

data (not obvious matches) to avoid inflating the evaluation performance.

We use the FWNN as a baseline for comparison, and experiment with two versions of our model: one uses a random forest classifier and the other uses a neural network classifier. For the random forest algorithm, we use 100 trees with a maximum depth of 10. We use the scikit-learn implementation with default values on the other parameter settings. For the neural network, we use three layers with an input layer of 42 nodes, an intermediate layer of 8 nodes, and an output layer of 1 node. Adam optimizer is used in this experiment. The auxiliary network for the rejection is also evaluated on top of our two models, and it has an identical structure to the neural network used for classification.

4.4.3 Results

Performance of Exact Rule Matching

The first result to look into is the performance of the matching rules that are extracted from the beam search. As these rules reveal the inner working of the company that provided the data for the experimentation, the exact details of the extracted rules cannot be revealed, but their overall performance can be discussed.

As shown in Table 4.1, we can observe that there is a dominating rule that can classify more than half of the test data (62.82% recall) with almost perfect precision (99.53%). The performance decreases a lot in the following rules

Table 4.3: Paired T-test on FWNN and our proposed models

Precision	RF	RF + TCP	NN	NN + TCP
80%	$1.185e^{-11}$	$2.574e^{-12}$	$1.279e^{-11}$	$1.179e^{-11}$
85%	$5.015e^{-9}$	$2.598e^{-10}$	$1.811e^{-9}$	$2.041e^{-9}$
90%	$1.542e^{-7}$	$1.326e^{-9}$	$2.364e^{-7}$	$3.595e^{-7}$
95%	$1.010e^{-2}$	$4.974e^{-4}$	$7.840e^{-4}$	$1.061e^{-4}$

in terms of recall. Thus, we keep only the first three matching rules in our experiments.

Similarity classification results

While the majority of the shipments have straightforward matches with previous shipments, a significant portion (33.68%) of shipments remain rather difficult to match using rules. In addition, it is necessary to remember there is 17% of all of the shipments belong to new importers and cannot be matched to any record. In these cases, we rely on machine learning models to provide a measure of similarity to aid us in the shipment match making process. We evaluated three different models on the remaining 33.68% of the data set where the matching rules do not apply. As a baseline model, we use a FWNN, which is the standard practice of the business that contributed the data. The second and third models are two versions of our framework wherein we train two types of classifiers with dynamic sampling: a random forest classifier and a neural network classifier. Both classifiers incorporate their own confidence network (See Section 4.3.5) trained so that it provides the confidence score used to reject samples.

From a business perspective, it is important to focus on part of the data where the model will return a prediction, as that is where the benefit comes from. Hence, a better model will have higher recall under the same precision. Figure 4.7 shows the precision-recall curve, which presents the trade-off between precision and recall as we change the rejection threshold τ from 0 to 1. We can see that both versions of our proposed model (random forest and neural network versions) can achieve a significantly higher recall under the same precision starting from a precision of 50%. The TCP rejection approach can improve the performance further. This translates to being able to process a significantly larger subset of shipments (by rejecting less shipments) while still satisfying precision requirements. The results for several representative precision requirements are shown in Table 4.2. In order to test the significant difference in those models, paired T-test has been conducted between FWNN

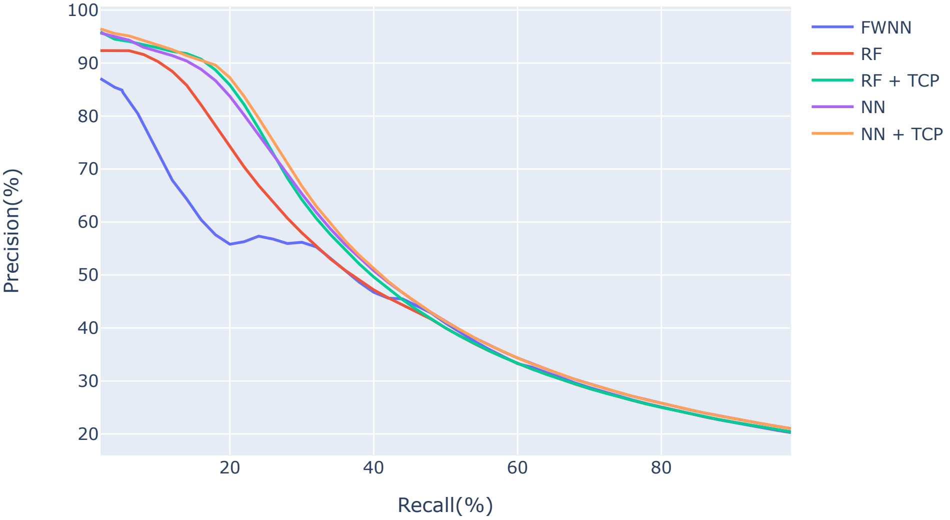


Figure 4.7: Precision-recall plot on test data.

and our proposed models. The results are shown in Table 4.3. All the p value are less than 0.05 which proved that there is a significant difference between FWNN and our proposed models.

In terms of time complexity, we applied K-D tree in the input to reduce the search space, and only do the pair-wise classification on the top 100 closest shipments, making the worst-case time complexity to be a constant $O(100)$ for a given input.

4.5 Conclusion

We propose a pair-wise classification method for the shipment importer prediction problem that outperforms the industrial standard FWNN model by a significant margin of precision across a wide range of recall values. Additionally, we extend the solution to a pipeline by introducing exact matching rules for the obvious importers to improve the time efficiency and an additional network that can better reject the classifier's prediction. This framework is significant also because it is applicable to similar problems and can in general terms be adapted to information retrieval domains.

From an algorithmic perspective, an interesting extension of this work can be exploring different confidence score representations and look into data cleaning techniques that can be applied to those nominal values.

Chapter 5

Confidence Range: Bridging Failure Detection and True Class Probability on selective hierarchical text classification

This paper focuses on selective classification for hierarchical text classification (HTC) problems. Selective classifiers reject unfavorable samples in a classification process, by employing a confidence score.

Two standard approaches to obtain a confidence score are to use an auxiliary confidence network to predict the classifier's True Class Probability (TCP) for each sample or apply Failure Detection (FD). This approach comes with the limitation that the confidence network pushes the confidence score to match the TCP even when a higher/lower confidence score would improve the performance.

We empirically show that TCP and FD do not always work well on HTC problems. We actually identify three data sets in which these classification schemes perform worse than the softmax output. To tackle this limitation, we propose a new confidence score called Confidence Range (CR) which gives the confidence network more flexibility to generalize to the samples, therefore allowing for a better performance in HTC problems. In our experiments we evaluate four different applicable rejection approaches built on top of long short-term memory (LSTM) and Transformer based seq2seq models. Our experiments show that our proposed method achieves the best performance among all the tested approaches in the selected HTC data sets.

This chapter is based on the following publication:

Chen, Xi and Bromuri, Stefano and Tan, Daniel Stanley. *2024*, DOI: 10.2139/ssrn.4244490, Confidence Range: Bridging Failure Detection and True Class Probability on Selective Hierarchical Text Classification, Applied Intelligence - Under review.

5.1 Introduction

Collecting and annotating classification data is a long and error-prone process. Parts of the data can be unclear and ambiguous, making them difficult to label. This is one of the reasons why the performance of a classification model does not always meet business requirements and expectations. In cases in which mistakes can be costly, it may be desirable to select the part of the data in which the classifiers perform well.

Selective classification frameworks use a confidence score indicating how reliable a model's predictions are for a specific input. A threshold can then be imposed on the confidence score to determine whether the classifier should reject a prediction or not. The goal of a selective classifier is to maximize the coverage (similar to recall) under the same cost requirements. Therefore, one may want to induce an ordering of the samples according to difficulty such that when a threshold is imposed, the correctly classified samples are above the threshold, and the incorrectly classified ones are below the threshold. One research direction is to learn from the classifier's output. Two main approaches that can exploit the classifier's output are Failure Detection (FD) [HDV18] and True Class Probability (TCP) [CTB$^+$19]. FD learns a separate failure score on top of an existing classifier. The failure score indicates whether the classifier can classify the sample correctly (emitting 1 as a score) or not (emitting 0 as a score). Figure 5.1 illustrates the FD target for a correctly classified sample. A potential issue of FD is that it optimizes the values towards the extremes (either 1 or 0), which as a result, leads to difficult samples with unintentionally high confidence scores, but should instead have lower confidence scores due to being close to a model classification boundary.

TCP, on the other hand, uses the classifier's predicted probability of the ground truth class as the target value to train the confidence network, as illustrated in Figure 5.1. Once trained, the confidence network tries to predict the true class probability for new samples. If the classifier is making a mistake, then the TCP will likely be low. Otherwise, the TCP is expected to be high. However, a problem arises on correctly classified samples: when the current confidence estimates are greater than the target true class probability, the training objective will force the confidence scores to be lower just to match the true class probability, despite the fact that a higher confidence score is better for correctly classified samples. Conversely, on incorrectly classified samples, when the confidence estimates are lower than the target TCP, the training objective will force the confidence scores to be higher despite being incorrectly classified.

To address these limitations, we propose Confidence Range (CR), which considers ranges in defining the learning objective for the confidence network instead of a single target value. For correctly classified samples, we optimize

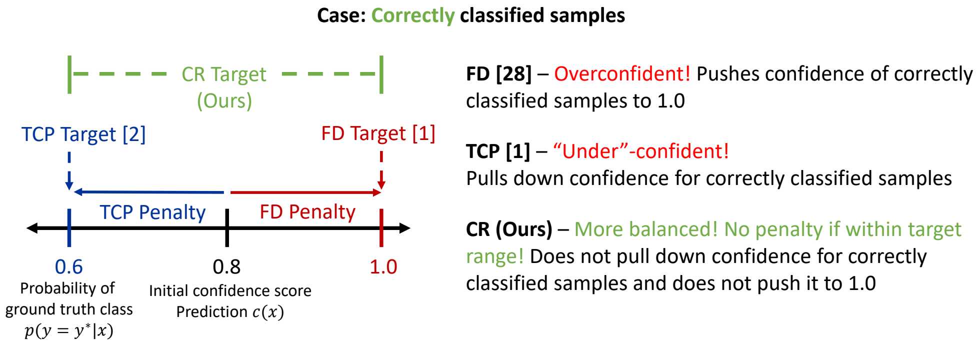

Case: Correctly classified samples
CR Target (Ours)
TCP Target [2]
FD Target [1]
TCP Penalty
FD Penalty
0.6
Probability of ground truth class $p(y = y^*|x)$
0.8
Initial confidence score Prediction $c(x)$
1.0
FD [28] – Overconfident! Pushes confidence of correctly classified samples to 1.0
TCP [1] – "Under"-confident! Pulls down confidence for correctly classified samples
CR (Ours) – More balanced! No penalty if within target range! Does not pull down confidence for correctly classified samples and does not push it to 1.0

Figure 5.1: Visualisation of the differences between True Class Probability (TCP) [CTB+19], Failure Detection (FD) [HDV18] and our proposed Confidence Range (CR) method.

the confidence scores to fall between the target range defined by the target TCP and the maximum value of 1 (FD target). For incorrectly classified samples, we optimize the confidence score to fall between the target range defined by the minimum value 0 (FD target) and the target TCP. This is illustrated in Figure 5.1. We stop penalizing the model once the output confidence score falls within our target confidence range. This removes the contradictory objectives and reduces mistakes with high confidence scores since we do not force the confidence scores toward the extreme values, and we do not force it to align with TCP target when it leads to better performance.

We would like to note that this approach is different from the range loss [ZFW+17] used in face recognition tasks. Range loss partitions the input samples while our confidence range partitions the prediction target.

In this paper, we focus on hierarchical text classification (HTC) [Gar83] as our case study in analyzing selective classification. HTC typically involves labels that have a hierarchical structure such as a hierarchy of classes organized like a tree, where the leaves of the tree are exclusive between each other. Rather than classifying only the leaves and transforming the problem into a multi-class problem, the presence of the super-classes can help specify better boundaries for the sub-classes [YSL+18]. In this contribution, we conduct our experiments on three different data sets presenting HTC labels: one private data set provided by DHL Express and two public data sets. Additionally, we also demonstrate the generalizability of our model to another data modality by evaluating on CIFAR-10 and CIFAR-100, which are benchmark datasets for image classification.

Our contributions are two-fold:

1. We propose Confidence Range (CR), a new objective for training the

confidence score. This achieves the best performance on three different data sets among all tested rejection approaches, demonstrating the effectiveness of our approach. Our simple approach improves upon TCP and can be a drop-in replacement to all models where TCP is applicable.

2. We report the first benchmark of threshold-based selective classifiers on the setting of HTC. To the best of our knowledge, this paper is the first to look into the HTC problem with a rejection option. We compare the performance of the selective classification incorporated in a LSTM-based NMT model and Transformer based model.

The structure of the paper is as follows: Section 5.2 covers fundamental related work concerning the HTC problem and selective classification; the LSTM-based NMT and Transformer models are explained in Section 5.3, together with all the rejection approaches; Section 5.4 will present the experiment and results; Section 5.5 concludes this paper by presenting our results and discussing future work.

5.2 Related work

This section presents the contributions that are mostly related to the one presented in this paper, concerning HTC and selective classification. Hierarchical classification literature can be categorized into three different types: flat classification, hierarchical local classification, and hierarchical global classification. The flat approach treats the task as a multi-class classification, instead, the hierarchical local approach implements multiple local models to determine next-level prediction, while the global approach uses one model and predicts all levels at once. For instance, Kowsari et al. [KBH$^+$17] proposed a deep learning model, which uses a neural network for each hierarchy, but with the development of deep learning, most of the recent research focuses on the hierarchical global approach [HCL$^+$19]. Wu et al. [WS17] and Gao et al. [Gao20] use fully connected networks to simulate the hierarchical structure. Wehrmann et al. [WCB18] and Huang et al. [HCL$^+$19] tested an LSTM-like network. Mao et al. [MTH$^+$19] simulate the decoding process with reinforcement learning. Lately, pre-trained Transformers [BB21] are dominating the scene in this domain. Most of the global models follow the seq2seq structure. Sequence generation for hierarchical text classification has proved its ability in Hierarchical classification [YSL$^+$18; US19]. Given the generalization capabilities of pre-trained Transformer models, these can be easily fine-tuned and adapted to deal with the HTC problem, while maintaining high performance [BB21].

Selective classification, also known as classification with rejection option, has been used in machine learning research for a long time [Cho57; Cho70]. There are two types of rejection options. One is a cost-specific approach, where we assume the cost for wrongly classified samples is known [DSV00; CDT$^+$95]. The other approach is threshold-based rejection option. This utilizes a confidence, trust, or reliability score [DSV00; JKG$^+$18] and compares it to a selected threshold to decide whether to accept the result. Our contribution falls under the variant using a threshold. Therefore, we mainly compare ourselves with threshold-based approaches. There are several model-specific approaches, such as SVM-based [FR02; BW08] and KNN-based. However, those are not easy to apply in a deep neural network(DNN) setup. K-nearest neighbors (KNN) approaches [Ati05; Dal09] are a natural choice for producing confidence scores with good explainability associated with the prediction. Some variations have been made in order to adapt KNN selective classification with deep neural networks (DNNs), including combining KNN with DNNs. The approaches that have been tried include: using softmax output to represent the weights for the KNN model [Ati05], applying KNN on a hidden layer on the DNN [MAN17], and applying KNN on all hidden layers and aggregating the results [LCA$^+$20]. Similarly, Jiang et al. [JKG$^+$18] used the distance ratio between the nearest class to the predicted class, while Lee et al. [LCA$^+$20] replaced the last layer with a KNN classifier. KNN-based approaches come with good performance and are easy to explain, but one of their biggest issues is scalability which makes them not suitable when it comes to production environments like in our case.

Other than KNN-based methods, another technique involves working on the softmax output directly. Vasconcelos et al. [VFB95] proposed a more reliable network by replacing the sigmoid activation function with a Gaussian activation function. Lately, Hendrycks et al. [HG16] and Geifman et al. [GE17] used the maximum value from softmax to represent model reliability and showed that the softmax output in DNN acts as a good baseline. Hecker et al. [HDV18] trained the model with additional output to predict whether the model would fail or not. Similar to Hecker's idea, instead of predicting the failure of the prediction model, Corbiere [CTB$^+$19] proposed an auxiliary confidence network to predict the sample's TCP.

One closely related area of research to the one of selective classification, is model uncertainty [APH$^+$21]. Currently, it is not clear how to infer uncertainty from reliability/confidence/trust scores. For a confidence score, the target is clear and it is to increase the coverage under the same risk [El-$^+$10]. However, for the model uncertainty, there is no single metric that can be used for measuring performance. Some common metrics, such as mean square error, are used for evaluating how well the model fits the data [GHV14], negative log-likelihood indicates how well-calibrated a model [Gal16] is, and the expected calibration error [GPS$^+$17] is used to score the calibration of maximum posterior predicted probabilities. As a consequence, the model uncertainty approaches and other calibration approaches [GPS+17; KLM19] will not be evaluated in this paper.

5.3 Methods

Problem Definition

In the following we will assume the training data $S_n = \{(x_i,\ y_i)\}_{i=0}^{n}$ is sampled i.i.d. from some unknown underlying distribution $P(X, Y)$. A selective classifier [GE17] is a pair of functions (f, g), where $f : X \to Y$ is a classifier that predicts class labels given an input, and $g : X \to \{0, 1\}$ is a selection function that determines whether to reject the prediction or not. The selective classifier is defined as follows:

$$(f, g)(x) = \begin{cases} f(x), & \text{if } g(x) = 1 \\ \text{reject}, & \text{if } g(x) = 0 \end{cases} \tag{5.1}$$

The selection function $g(x)$ is dependent on a confidence score $C(x)$ and a threshold τ, defined as follows:

$$g(x) = \begin{cases} 1, & \text{if } C(x) \geq \tau \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

In the selective classification literature, the threshold is defined based on either risk or precision requirements [JKG+18; GE17]. In this paper, we use precision since it is more suitable for a business perspective. For a selective classifier, the goal is not to maximize the overall accuracy. Instead, the goal of a selective classifier is to find the model $f(x)$ and the selective function $g(x)$ that has the highest coverage under the same precision requirements, where coverage refers to the percentage of data that is not rejected by the selective function.

5.3.1 Classification Model

For the HTC problem, we are validating two commonly used deep neural networks: an LSTM-based NMT model and a Transformer. To maintain similar notations in the literature, all variable notations in this section are only applicable within their own sub-section.

Long Short Term Memory Based Neural Machine Translation

An NMT encodes all the input symbols into a hidden space, then predicts the output symbols one by one. Formally, the model transforms the sequence vectors $x = (x_1, x_2, ..., x_j)$ into sequences of vectors $y = (y_1, y_2, ..., y_k)$, where j and k are the length of the input and output respectively. The transformation is performed by the LSTM unit in two phases: encoding and decoding process. For the encoding process, the LSTM unit will read the tokens one by one and output a Context Vector $c = q(f(h_1, ..., h_j))$ where f and q are non-linear functions, h is the hidden state. The context vector encodes all the information of the sequence.

$$p(y) = \prod_{t=1}^{m} p(y_t | y_1, y_2, \ldots, y_{t'-1}, c) \tag{5.3}$$

The attention mechanism has been first proposed by Wang et al. [WHZ+16]. It allows a neural network to concentrate on different parts of a sentence that may be more important toward a specific machine learning task. Let $H = \{h_1, h_2, ..., h_j\}$, V_α represents the importance of the hidden states. The final context vector c is calculated as follows:

$$\begin{aligned} c &= \tanh(W_p r + W_x h_j) \\ r &= H\alpha^T \\ \alpha &= \text{softmax}(W_T M) \\ M &= tanh\big(\big[W_h H + W_v v_a \otimes e_N\big]\big) \end{aligned} \tag{5.4}$$

where r is a weighted representation of the sentence. h_n is the final output from the LSTM. e_N is a column vector with 1s, the function of $v_a \otimes e_N$ is to repeatedly concatenate v for N times. W_p, W_x, W_T, W_h and W_v are learnable weights of the LSTM.

The idea of attention has been further explored in [MLZ+17; CMP+21] and we refer the interested reader to these publications for a complete discussion on attention models.

Transformer

The Transformer model is purely attention-based. The variants of the Transformer have been successfully used in NLP tasks [HLG+20; LOG+19]. The essence of the Transformer is the multi-head attention (self-attention) mechanisms. With respect to the basic attention mechanism, multi-head attention has the benefit of being parallelizable, but also that it can remember longer-range dependencies happening in sequential data.

The Transformer model consists of an encoder and a decoder, sharing a similar structure. Both consist of multiple identical blocks. Each block has multi-head attention and a feed-forward network that connects with a residual connection [HZR+16] in the sub-layer.

Multi-head attention consists of multipless scaled dot-product attentions. Each independent attention output is then concatenated and linearly transformed into the expected dimension. More formally, we can define multi-head attention as follows:

$$\text{Multi-head}(Q, K, V) = [head_1, head_2 ... head_h] * W^o$$

Where each $head_i$:

$$\begin{aligned} head_i &= \text{Attention}(QW_i^Q, KW_i^k, VW_i^V) \\ &= \text{softmax}\Big(\frac{QW_i^Q * KW_i^{kT}}{\sqrt{(d_k)}}\Big)VW_i^V \end{aligned} \tag{5.5}$$

where h is the number of heads, $d_k = d_{model}$/h, d_{model} is the dimension of the output. The attention function is computed as a set of queries, packed in the matrix Q, with respect to a set of keys and values, packed in a K and V matrices. W^o, W^Q, W^k, W^V are parameter matrices.

5.3.2 Rejection options

In selective classification, whether a test sample is rejected or not depends on its confidence score. In our experiments, we test four different confidence scores: Maximum Softmax Probability (MSP) [HG16], Failure Detection (FD) [HDV18], True Class Probability (TCP) [CTB+19], and our proposed Confidence Range (CR).

Maximum Softmax Probability (MSP)

One of the simplest, yet effective, approaches to represent a sample's confidence score, is to use the maximum value from the softmax output of the neural network. It has been shown that the prediction probability of incorrect samples tends to be lower than the prediction probability for correct samples [HG16; GE17]. Geifman et al. [GE17] and Corbiere et al. [CTB+19] have shown that MSP is a strong baseline for selective deep learning classifiers.

For hierarchical classification with K levels, the softmax probability is the multiplication of probability on all of the hierarchical levels:

$$P(Y|X) = \prod_{i=1}^{K} P(Y_i|X) \tag{5.6}$$

The confidence score is the maximum value from softmax probability: $\max(P(Y|X))$.

Failure detection (FD) & True Class Probability (TCP)

MSP is a common baseline for representing the confidence score. However, it has been shown to be overconfident in some cases [GG16]. FD and TCP specify an alternative way to represent the confidence score.

Both approaches share a similar idea: they learn the confidence score from the classifier's prediction. The main difference between the two is that FD specifies a binary target whose value depends on whether the classifier can predict the sample correctly (1) or not (0), whereas TCP predicts the probability of the ground truth label, given the classifier's prediction. For any sample pair $\{x_i, y_i\}$, a classifier is trained first, which outputs the predicted probability over each class $p(y|x_i;\theta)$, where θ denotes the parameters of the network. Then, an additional confidence network is trained to learn FD / TCP scores on training samples. For TCP, the learning target is the classifier's probability on the sample's ground true label, which is $p(y = y_i|x_i;\theta)$. As for FD, the learning target is 1 if the classifier can classify the sample correctly, 0 otherwise.

After the confidence network is trained, it is then used to predict the new sample's confidence score. Ideally, the correctly classified sample will have a high confidence score, while the incorrectly classified sample will have a lower confidence score. By setting a threshold to reject certain samples where the model most likely misclassifies, we can achieve high precision on the rest of the data.

Confidence Range (CR)

In order to improve the confidence network's generalizability with respect to TCP, we propose the following changes: first, we use the softmax output from the classifier as an additional feature input for the confidence network when predicting the target. Second, we propose Confidence Range (CR) as a new learning target that bridges the gap between FD and TCP. Figure 5.2 shows the structure of our proposed confidence network.

As shown in Figure 5.1, for the correctly classified sample x_i, the TCP target is equal to the maximum value from the softmax output:$p(y = y_i|x_i) = \max(f(x_i;\theta))$. When the predicted score is larger than $\max(f(x,\theta))$, the confidence network still pushes the predicted value to $\max(f(x,\theta))$. However, this optimization direction is not good for performance, since in a correctly classified sample it is better to have a higher confidence score. As a consequence, we propose a new learning target CR that combines the ideas of FD and TCP. In our proposed CR, we set the loss to zero if the predicted confidence score falls in the designated range during the training. For the correctly classified sample, the learning target range is between $[p(y = y_i|x_i)], 1]$, and for misclassified

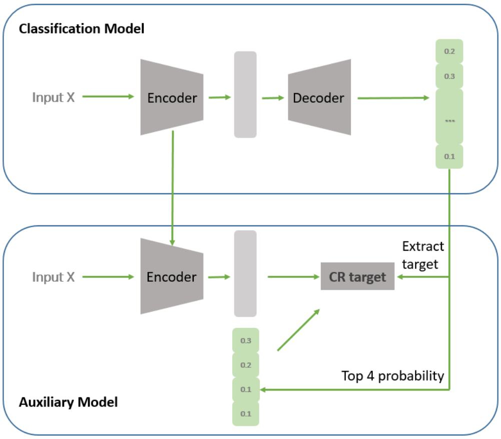


Figure 5.2: Overview of our Confidence Range architecture. After training a classifier, we train an auxiliary confidence network and use the classifier's output to define our target confidence range.

samples, the target range is between $[0, p(y = y_i|x_i)]$. More formally:

$$L_{CR} = \frac{1}{N}\sum_{i=1}^{N} \max\Big(0, s(y_i, \hat{y}_i) \times \big(p(y = y_i|x) - c(x)\big)\Big)^2 \tag{5.7}$$

$$s(y, \hat{y}) = \begin{cases} 1 & \text{if } y = \hat{y} \\ -1 & \text{if } y \neq \hat{y} \end{cases} \tag{5.8}$$

where $\hat{y}$ is the prediction and $c(x)$ the output of the confidence network. This allows the auxiliary confidence network to predict any value inside the range without a penalty. The main advantage of this range target compared to TCP is that it gives the auxiliary confidence network more generalization flexibility in learning a suitable confidence score ordering of the data within the target range as opposed to TCP which forces it to follow a fixed target defined by the classifier. This flexibility also helps in improving the coverage compared to the TCP approach as we show in our experiments (Section 5.4).

Since the confidence network is learning from the classification model, it would be useful to integrate the classifier's output when predicting the confidence score. Similar to Blatz's work [BFF$^+$04], the classifier output is concatenated with the hidden layer when predicting the confidence score. Due to the complexity of the hierarchical structure of the problem, it is not realistic to predict all the possible outcomes over the whole hierarchy. As a compromise, we only use the top four highest predicted probabilities as additional input for the auxiliary confidence network, with a beam search having a width of three.

5.4 Experiments

5.4.1 Data set

Three different data sets have been tested in our experiments: a private data set provided by DHL and two public data sets: Amazon Product description [1] and Amazon Review data from Kaggle competition. The private data set can not be released due to the confidential agreement with DHL.

DHL Shipment Description. The data used in this experiment were collected over a period of ten months and it concerns shipments moving towards one destination country via the DHL network. The data set contains the following features: item description, origin, destination, and harmonized system code (HSCode) [CBV21]. The HSCode is dependent on the shipment content, and it is directly linked with import/export tariff. The model is trained based on the description, origin, and destination to predict the HSCode associated with the shipment. For example, if the shipment description is ”Used Aluminum door frame”, its corresponding HSCode is 76.02.00.

Amazon Product Description. This data set is a collection of product descriptions from Amazon. Each description has been classified into three sub-levels. In this experiment, we used data from three super-categories: Arts Crafts and Sewing, CDs and Vinyl, and Electronics. All the combinations of labels in the three levels that have less than 10 samples are removed during pre-processing.

Amazon Product Review. This data set is a collection of product reviews from Amazon. It contains a product categorization expressed in a hierarchical structure and its review from the buyer. We use the review content to predict the product category.

5.4.2 Implementation details

All the chosen data sets have three hierarchical levels. We apply 10-fold cross-validation with validation and test set. For the DHL Shipment Description data set, we order the shipment in time, then we use a rolling window evaluation approach to prevent information leakage. For the Amazon Product description and Amazon product review, we use a stratified split with an 80%, 10%, 10% partitioning for train, validation, and test sets respectively. The size of the data is shown below, the number of samples for DHL data set is the average in the rolling window.

We use a bidirectional-NMT model and a Transformer model as basic classifiers. Both models are trained with a teacher-forcing technique [WZ89]. The same classifier is used in evaluating different rejection approaches. BERT

---

[1] `https://jmcauley.ucsd.edu/data/amazon/`

Data	Train	Val	Test
DHL Shipment Description	925,168	115,647	125,627
Amazon Product Description	1,114,445	139,306	139,306
Amazon Product Review	40,000	5,000	5,000

Table 5.1: Data set statistics

[KT19] embedding is used in this experiment. In the NMT model, the dimension of the LSTM units is 100 with a dropout of 0.2 since we noticed during the experiment that it tends to over-fit with larger unit sizes. Rmsprop is used as the optimizer with a learning rate of 0.001. Concerning the Transformer, it contains two layers of Transformer blocks, with 4 multi-head attentions. The dimension of the model is 768. The optimizer is Adam with $\beta_1 = 0.9, \beta_2 = 0.98$, $\epsilon = 1e^{-9}$ with scheduled learning rate similar to [VSP+17]. As for the optimizer, we use RMSprop optimizer starting with a learning rate at 0.0001 with a weight decay of 0.975 per epoch except for the Amazon Product Review data set, which uses 0.0005 as the initial learning rate and 0.8 as the weight decay.

5.4.3 Results and Discussion

In this contribution, both NMT and Transformer are evaluated on four different rejection approaches: Maximum Softmax probability (MSP), Failure detection (FD), True class probability (TCP), and Confidence Range (CR).

When evaluating the different rejection approaches on the same classifier, we choose four representative precisions that are normally used in the industry for coverage analysis, which are 80%, 85%, 90%, and 95% respectively. We are comparing the coverage under each precision. The table below is the overview of coverage under those chosen precisions. The way to calculate the coverage is to order the samples by their confidence score first, then move the confidence score threshold gradually, until the precision on the rest of the data is equal to the desired precision. The coverage is the percentage of the data where its confidence score is larger than the threshold.

We report the performance values as the average of a 10-fold cross-validation. The statistic tests are discussed in this section. For the results analysis, we are interested in two aspects: a) The coverage performance for different rejection approaches; b) The performance of NMT and Transformer under the same rejection approach.

Rejection Approach Evaluation. In Fig. 5.3 we plot the difference between MSP and other rejection approaches. Based on the plot, we can conclude that the MSP is a strong baseline also in the HTC with a rejection option. FD and TCP do not seem to perform well in sequential tasks, despite

		NMT				Transformer			
	Precision	MSP	FD	TCP	CR	MSP	FD	TCP	CR
Shipment Description	80%	35.21	32.06	33.29	**36.74**	38.22	34.46	35.82	**39.28**
Shipment Description	85%	26.92	24.39	25.64	**28.84**	29.51	25.61	27.16	**31.51**
Shipment Description	90%	18.47	17.22	18.43	**20.97**	20.31	16.74	18.30	**22.49**
Shipment Description	95%	9.08	8.70	9.83	**11.14**	10.01	7.53	8.89	**11.56**
Product Description	80%	**81.60**	80.55	80.68	81.45	**82.84**	81.86	82.03	82.73
Product Description	85%	72.80	69.89	70.26	**72.88**	73.91	71.91	72.33	**74.08**
Product Description	90%	62.71	56.85	57.45	**63.18**	63.71	60.97	61.81	**64.20**
Product Description	95%	47.99	39.43	40.62	**49.08**	48.78	45.58	47.48	**50.42**
Product Review	80%	37.30	30.32	30.39	**38.02**	38.30	32.02	33.74	**38.96**
Product Review	85%	28.06	22.65	22.99	**28.94**	28.86	24.20	25.09	**30.49**
Product Review	90%	19.25	14.18	14.58	**20.89**	20.14	17.31	18.73	**21.51**
Product Review	95%	10.46	7.47	7.54	**11.64**	11.44	9.11	10.59	**12.92**

Table 5.2: Selective classification coverage results on the three selected data sets, using precision to specify the rejection threshold.

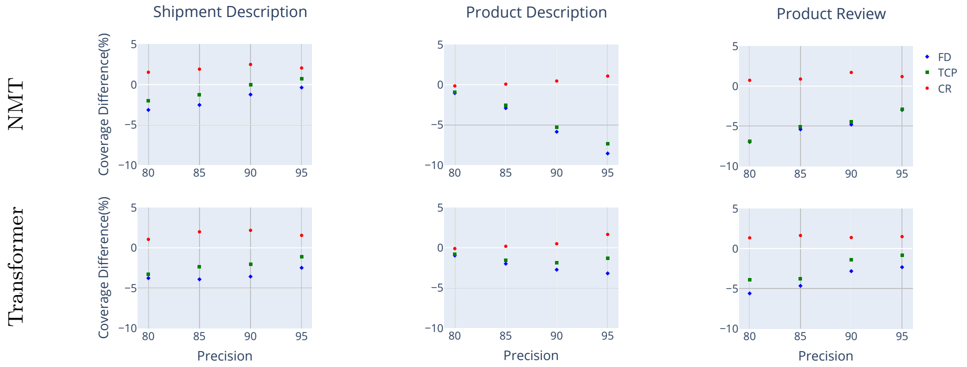


Figure 5.3: Rejection option comparison: X is the precision, Y is the improvement in terms of coverage, expressed as coverage(Evaluation approach) - coverage(MSP). A Negative value means that the approach is worse than the baseline.

	Precision	NMT(MSP VS CR)	Transformer(MSP VS CR)
Shipment Description	80%	$7.10e^{-6}$	$8.47e^{-5}$
Shipment Description	85%	$7.78e^{-7}$	$3.02e^{-6}$
Shipment Description	90%	$3.16e^{-9}$	$2.67e^{-8}$
Shipment Description	95%	$5.20e^{-8}$	$9.62e^{-6}$
Product Description	80%	$5.50e^{-3}$	$4.24e^{-3}$
Product Description	85%	$1.48e^{-1}$	$6.96e^{-4}$
Product Description	90%	$2.45e^{-5}$	$7.09e^{-6}$
Product Description	95%	$4.55e^{-6}$	$5.57e^{-6}$
Product Review	80%	$2.93e^{-2}$	$6.00e^{-3}$
Product Review	85%	$1.52e^{-1}$	$9.31e^{-4}$
Product Review	90%	$3.40e^{-2}$	$9.20e^{-4}$
Product Review	95%	$1.20e^{-1}$	$6.35e^{-3}$

Table 5.3: The p-value of Student's t-test on MSP and CR approach.

working well in image-related tasks [HDV18; CTB+19]. One possible cause could be the different classifier architecture used in this paper. Considering that the classifier in this problem is a seq2seq model, this implies that when creating the auxiliary confidence network, we can only use up until the hidden layer to predict TCP / FD, and the decoder knowledge is not utilized.

The second observation is that CR achieved the best performance among most of the tested approaches except for the Amazon Product description data set. CR performs slightly less than softmax when the precision threshold is 80%. We can notice that the difference between CR and MSP is rather small when the precision threshold is close to the accuracy. The accuracy in this data set is around 72%. But the coverage difference is increasing along with the precision threshold, as shown in Figure 5.3

In order to validate whether there is a significant difference between CR and MSP, we applied the Student's t-test on each precision threshold. The results are shown in Table 5.3

The result shows that for the Transformer, all the p-values are less than 0.05, which proves there is a significant difference between CR and MSP. For the LSTM-based NMT model, our model performs better on average but it is not statistically significant in certain thresholds. Especially in the Amazon product review data set. We noticed that the training process is not stable in some splits, the data size (40k) may have an impact on it. This result may also be due to the limitations of the model in identifying important words as compared to the Transformer which uses multi-head attention.

LSTM-based NMT VS Transformer. Other than comparing different

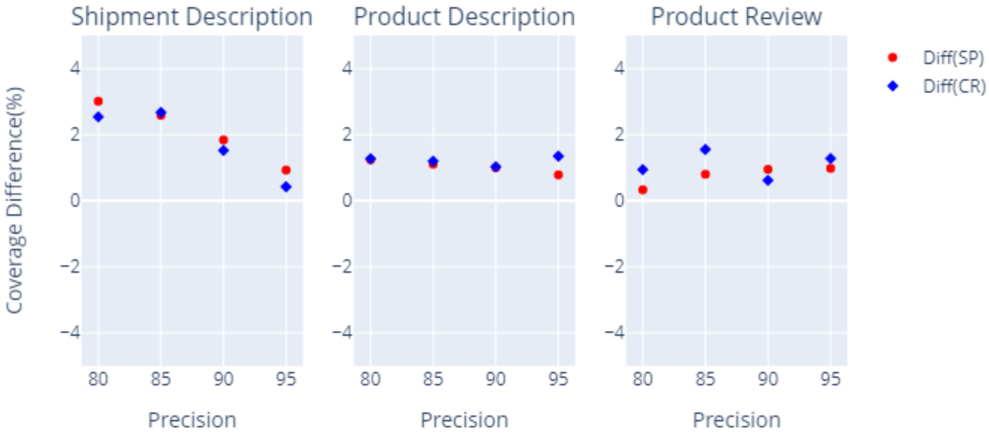


Figure 5.4: Comparison between LSTM-based NMT and Transformer for both MSP and CR option. In this plot, the diff is the Transformer's coverage minus the LSTM-based NMT's coverage.

rejection approaches on the same classifier, we also looked into the comparison of the same rejection approaches on different classifiers. As aforementioned, the CR approach achieves the best performance in most of the cases while FD and TCP are worse than the baseline MSP in general. Hence, only the MSP and CR techniques are compared here.

The comparison is shown in Figure 5.3. It can be easily concluded that the transformer consistently outperforms the LSTM-based NMT model on both rejection techniques, though there is no clear signal that Transformer gained more coverage from the CR technique.

5.4.4 Ablation study

Compared with the original TCP approach, our proposed Confidence Range (CR) has made two main changes: 1. Change the TCP target to a more relaxed CR target. 2. Integrate the softmax output as additional input when predicting the CR. We conduct an ablation study to better understand the effects of these changes.

When looking at the result as shown in Table 5.4, on all three data sets, our proposal has the best performance compared with the baseline TCP approach. It is obvious to see that when integrated with the MSP output, the performance increases a lot. Changing the target to our confidence range further improves the performance even further.

Though, the difference in one of the datasets is rather small between TCP (with MSP) and Confidence Range (with MSP). In order to understand whether there is a significant difference between those two approaches, we also conduct

		Loss target		MSP output		Shipment Description				Product Description				Product Review			
		TCP	CR	with	without	80%	85%	90%	95%	80%	85%	90%	95%	80%	85%	90%	95%
NMT	Baseline	✓			✓	33.29	25.64	18.43	9.83	80.68	70.26	57.45	40.62	30.39	22.99	14.58	7.54
NMT			✓		✓	34.23	26.38	18.88	10.12	81.01	71.25	59.36	43.62	33.81	25.22	17.41	9.39
NMT		✓		✓		36.45	28.65	20.67	10.79	81.51	**72.88**	63.08	48.92	37.47	28.85	20.37	10.97
NMT	Ours		✓	✓		**36.74**	**28.84**	**20.97**	**11.14**	**81.45**	**72.88**	**63.18**	**49.08**	**38.02**	**28.94**	**20.89**	**11.64**
Transformer	Baseline	✓			✓	35.82	27.16	18.30	8.89	82.03	72.33	61.81	47.48	33.74	25.09	18.73	10.59
Transformer			✓		✓	36.60	27.76	19.09	9.38	82.31	72.84	62.29	48.14	35.14	26.46	19.57	10.98
Transformer		✓		✓		38.79	31.01	21.88	10.93	82.73	74.04	64.13	50.08	38.41	29.93	21.10	12.28
Transformer	Ours		✓	✓		**39.28**	**31.51**	**22.49**	**11.56**	**82.73**	**74.08**	**64.20**	**50.42**	**38.96**	**30.49**	**21.51**	**12.92**

Table 5.4: Results of the ablation study where we add or remove different components of our model as well as its effects on the two different model architectures.

		NMT		Transformer	
	Precision	TCP VS CR	TCP(w. MSP) VS CR(w. MSP)	TCP VS CR	TCP(w. MSP) VS CR(w. MSP)
Shipment Description	80%	$1.04e^{-4}$	2.66^{-2}	$2.29e^{-5}$	$5.12e^{-3}$
Shipment Description	85%	$5.87e^{-4}$	$2.65e^{-1}$	$4.87e^{-2}$	$2.13e^{-2}$
Shipment Description	90%	$1.76e^{-4}$	$3.56e^{-2}$	$9.12e^{-4}$	$3.32e^{-2}$
Shipment Description	95%	$2.48e^{-2}$	$1.05e^{-2}$	$4.75e^{-3}$	$2.94e^{-2}$
Product Description	80%	$3.93e^{-4}$	$2.68e^{-1}$	$8.18e^{-3}$	$9.44e^{-1}$
Product Description	85%	$1.28e^{-3}$	$8.49e^{-1}$	$7.43e^{-4}$	$1.53e^{-1}$
Product Description	90%	$1.57e^{-5}$	$7.25e^{-2}$	$5.12e^{-3}$	$1.76e^{-1}$
Product Description	95%	$4.28e^{-3}$	$3.81e^{-3}$	$3.13e^{-4}$	$2.37e^{-2}$
Product Review	80%	$1.13e^{-4}$	$1.58e^{-1}$	$2.97e^{-4}$	$1.77e^{-2}$
Product Review	85%	$1.42e^{-3}$	$9.09e^{-1}$	$2.23e^{-3}$	$1.80e^{-2}$
Product Review	90%	$7.51e^{-3}$	$6.55e^{-1}$	$1.71e^{-3}$	$1.98e^{-2}$
Product Review	95%	$1.54e^{-3}$	$8.84e^{-1}$	$1.28e^{-3}$	$2.91e^{-2}$

Table 5.5: The statistical significance of the performance differences in our ablation study.

a t-test on each pair and summarize the results in Table 5.5.

For both NMT and Transformer models, the TCP is statistically different from the CR approach on all datasets. However, when we add the softmax, the TCP(with MSP) approach is closer to the CR(with MSP) approach. Both models perform differently. We would like to note that including MSP is part of our modifications and not part of the original approach of TCP [CTB+19]. For the NMT model, those are no significant differences in most of the precision thresholds. As for the Transformer, there is a significant difference in the Shipment description and product review data sets. For the Product Description data set, there is only a difference in high precision thresholds. The difference between the Product Description data set and the rest is that both models have high accuracy when predicting the target which is around 75%.

In summary, our experiments show that our CR approach is better than the TCP approach for the tested selective HTC problems. Adding softmax as additional input to predict TCP can improve the model performance.

	Dataset	AUPR-Success (↑)	AUPR-Error (↑)	AUC (↑)	FPR-@95-TPR (↓)
TCP [CTB+19]	CIFAR-100	92.68%	**73.68%**	86.28%	62.96%
CR (Ours)	CIFAR-100	**94.10%**	72.06%	**87.20%**	**62.36%**
TCP [CTB+19]	CIFAR-10	**99.24%**	**49.94%**	**92.12%**	44.94%
CR (Ours)	CIFAR-10	99.23%	47.80%	91.95%	**44.23%**

Table 5.6: Performance comparison on image classification data (CIFAR-10 and CIFAR-100).

5.4.5 Evaluations on image data

For completeness, we also evaluated our method on CIFAR-10 and CIFAR-100, which are two benchmark datasets for image classification. We follow the evaluation setup of Corbiere et al. [CTB+19] and report the area under the precision-recall curve using the correct class as the positive class (AUPR-Sucess) and its counterpart that uses the errors as the positive class (AUPR-Error). We also report the area under the receiver operating characteristic curve (AUC) and the false postive rate at 95% true positive rate (FPR-@95-TPR). Table 5.6 shows the results in comparison to TCP [CTB+19]. Our proposed CR outperform TCP [CTB+19] across all the metrics on CIFAR-100 and performs competitively on CIFAR-10. This demonstrates that our method is not only effective on hierarchical text classification but also generalizes to image classification tasks.

5.5 Conclusion

In this paper, we proposed a new confidence score called confidence range (CR) for selective classification. The results are compared under the context of three HTC problems. The takeaways from this paper are: a) Our CR method achieved the best performance among all tested rejection methods in the HTC problems selected. b) FD and TCP do not seem to always work well in selective HTC problems, against MSP as a baseline. c) transformer is a better option than basic LSTM-based NMT when it comes to selective HTC problems. We also demonstrated the capabilities of our model in image classification, showing the generalizability of our approach to other data modalities.

In terms of future work, CR is still a two steps method. From a feedback control theory perspective, it is possible to integrate information concerning the confidence of the model during the training and change the training loop of the neural networks. In this contribution, we also did not put emphasis on how to sample the data in the best way to train on the rejection approach. As it is clear that part of the data will never be covered when it comes to selective classification problems, certain sampling strategies might lead to further improving the results shown in this contribution.

Chapter 6

A transfer learning approach to predict shipment description quality

International shipments always have a harmonized system code (HSCode) associated with them, to determine the tariff for the custom declaration. The HSCode is derived from the shipment description that the customer provides, which makes the quality of the description important to assign the correct code. When the description is too generic or incomplete, the logistic company will have to contact the customer in order to find out the content of the shipment. Due to the fact that there is no effective way to identify the quality of description, we developed a description quality evaluation model, based on deep learning combined with domain knowledge. By using a 2000 shipments data set with scores ranging from 0 to 4 provided by experts, where 4 represents the best quality possible, the developed model can classify 45.17% of the data correctly and 43.95% of the data with 1 score difference (i.e predict label 1 as 2 or 0) from the human annotated ground truth. This model can be used for historical data analysis, and potentially giving customers on-site feedback when they are providing a bad description for the shipment content.

This chapter is based on the following publication:

Xi Chen, Stefano Bromuri, and Marko Van Eekelen. *2022*, DOI: 10.1145/3477314.3507339, A transfer learning approach to predict shipment description quality, In Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing (SAC ’22).

6.1 Introduction

The shipment description is important information when someone ships an international package, as it is used for determining the Harmonized System Code(HSCode) and the tariff that shippers have to pay. Despite their importance, there currently are no effective ways to identify bad descriptions that would cause the authority to react. To address this issue, we developed a description quality evaluation algorithm.

We started from a definition of quality that revolves around how well the description allows to classify the shipment description into a HSCode. Based on the different amounts of information it carries, the description quality is defined in five different categories: Really Bad (0), Bad (1), Average (2), Good (3), and Really Good (4). We used the following features to define our quality score associated with the shipment descriptions:

- Model Reliability: A transfer learning feature obtained from the output of a neural network. The neural network is pre-trained with an HSCode classification task.
- Domain Knowledge: A vector that records which functional properties of the object are available in the description
- Linguistic Feature: Features that concern the linguistic structure of the shipment description.

With the help of domain experts, we labelled 2000 items for training a machine learning model and 500 items for testing. We trained an ordinal regression model [Ped15] on top of the features described above. The model can achieve accuracy at 45.17%, and 43.95% of the data with 1 score difference.

The contribution of the paper is to illustrate how the combination of transfer learning and domain knowledge can be used to obtain a quality score concerning a text description of a product in an automated text scoring(ATS) problem. This approach can be easily transferred to other ATS problems like determining the quality of the product review.

The structure of the paper is as follows: The related work will be discussed in Section 6.2. In Section 6.3, we will explain the data that is used in this experiment. The methodology will be introduced in Section 6.4. Section 6.5 and 6.6 are for the results and conclusion respectively.

6.2 Related work

Classifying the description quality into different score/categories is an ATS problem. We can find three main categories of ATS problems related to our work, that are Text Readability, Machine Translation Quality, and Automated Essay Scoring.

6.2.1 Text Readability

Readability is a common approach to assess the quality of written text. Informally, text readability is understood as the easiness at which a human can comprehend a given text.

Numerous features and approaches exist towards predicting text readability. Lexical syntax [DK82], discourse connectives [MPJ+04] and cohesive features [HH14] are examples of hand-crafted features that have been used in combination [PN08; FJH+10; CSD+17] to assess text readability.

Thanks to the fast development of deep learning, embedding-based solutions have become the predominant paradigm to tackle the assessment of text readability. Cha et al. [CGK17] and Deutsch et al. [DJS20] use the feature that extracted from the embeddings. Additionally, some approaches apply Recurrent neural network (RNN) [NO18; AP19] and Transformer[MPR21] on top of the embeddings.

6.2.2 Machine Translation Quality

Another closely related domain is Machine Translation Quality Estimation. From a technical perspective, it is very close to performing a readability prediction. The features that are used in readability, such as coherence, semantic features and other linguistic features, are also often used to evaluate the machine translation quality [GAS05; LB+05; STC+09]. The idea of embeddings and deep learning models are also dominating the current state of the art in this task. Kim et al. [KL16] used a bidirectional RNN in addressing this problem. Shah et al. [SBB+16] proposed to use hand-crafted features, output from neural machine translation (NMT), and the average of word embedding to predict the quality of a translation. Similar to Shah's approach that learns from the translation model, Kim et al. [KLN17] proposed a two-steps approach: train the word prediction and the quality prediction with stack propagation. Following Kim's work, Li et al. [LXC+18] tried to combine two stacked models into a single one. Instead of using a predictor-estimator approach, Cui [CHL+21] pre-trained the evaluation model on generated data that is close to the test data set.

6.2.3 Automated Text Scoring

Different the readability and machine translation quality evaluation problem, essay scoring is in general a more complex problem, because it requires good understanding on essay clarity, topic development, and persuasiveness [KN19].

The neural network based solutions have been proposed by Taghipour et al. [TN16] and Alikaniotis et al. [AYR16] which output-performed the hand-crafted solutions. Taghipour et al. used a Convolutional recurrent neural network while Alikaniotis et al. used a Long Short term memory network (LSTM). Both models are applied on the word level. In order to make the model faster at learning, Dong et al. [DZY17] designed a hierarchical model where first a convolution is applied on the sentence to retrieve the sentence vector, and then each sentence vector is connected into a sequence to be put in input to an LSTM.

Some researchers argued that deep learning networks are incapable of capturing all the information due to the inefficiency of the structure. To deal with the limits of coherently representing the global meaning of the essays, Tay et al. [TPT$^+$18] and Farag et al. [FYB18] included auxiliary neural coherence features in their deep learning models.

End-to-end solution are becoming predominant in ATS problems. Our problem can also be addressed by using an end-to-end model if a large dataset can be provided. However, in general, obtaining a large dataset is difficult. That is why we propose to use transfer learning. This enables us to train a sufficiently accurate model with a small dataset.

6.3 Data

The quality target that we are going to predict is defined in five different categories depending on the amount of information it carries in classifying HSCode. The labels are: Really Bad (0), Bad (1), Average (2), Good (3), and Really Good (4) respectively. The higher score indicates more information contained in the description.

We randomly sampled 2000 historical shipment descriptions from historical data that available at DHL as training data set and 500 descriptions as test data set. Also, one million historical shipment descriptions and their related HSCodes also got collected. The purpose of the one million descriptions is to train a hierarchical classification model which can be used for extracting the model reliability. The basic statistics concerning the length of the descriptions can be found in Table 6.1. The data was being collected in early 2021.

In order to get the labeling data, four domain experts were asked to label the quality of the description for the 2500 collected data items aforementioned. Since it is a subjective score and it requires domain knowledge, we used the majority vote to determine the final label. The description was removed if the maximum difference in grade between the domain experts was larger than 2 quality grades. The label distribution is shown in Figure 6.1.

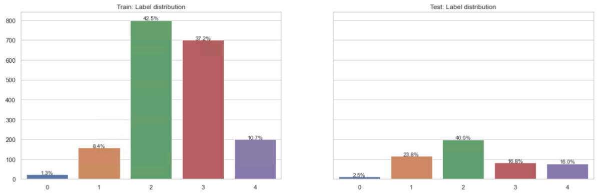


Figure 6.1: Label distribution

	Min length	Max length	Median	Average
Train	1	28	3	4.28
Test	1	17	3	3.95
Historical Data	1	63	3	4.10

Table 6.1: The statistics of the dataset

6.4 Methodology

In order to create a generalized model with limited labeled data, we used several hand-crafted features for classification instead of building an end-to-end model. The features that are used are model reliability, domain knowledge, and linguistic features. The architecture of the model is shown in Figure 6.2.

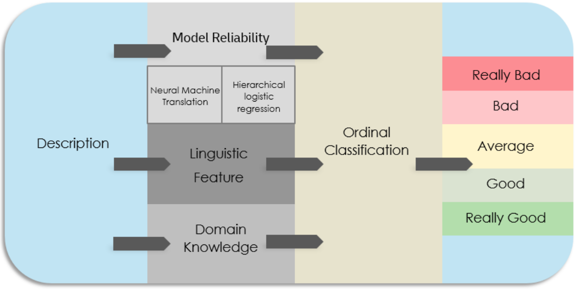
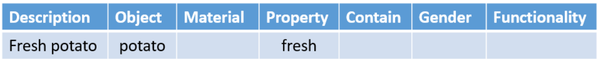


Figure 6.2: Model architecture

6.4.1 Model Reliability

The first feature used to model our quality score is model reliability. This feature is based on transfer learning knowledge obtained out of a NMT that translates shipment descriptions in HSCode. The model reliability score indicates how reliable the model's prediction is. Intuitively, a bad quality description will lead to a lower reliability score. It has been proven that the reliability score can be used as an indicator for detecting outlier or mis-classification [HG16].



Description	Object	Material	Property	Contain	Gender	Functionality
Fresh potato	potato		fresh			

Figure 6.3: Illustration of domain knowledge. The description will be converted to a feature vector with binary value $[1, 0, 1, 0, 0, 0]$.

In our experiment, we use the maximum softmax output [GE17; HG16] to represent the model reliability.

In order to obtain the reliability score we trained a classification model with historical data {Description, HSCode}. The HSCode classification problem can be considered a hierarchical text classification (HTC) problem. The NMT model has proven its ability in tackling HSCode prediction problems [CBV21]. In this contribution, we applied two models to extract the model reliability score: Hierarchical Logistic Regression(HLR) and Neural Machine Translation(NMT).

6.4.2 Domain Knowledge

The issue of model reliability is that it might get biased by the historical data. In order to alleviate this problem, we introduce domain knowledge as a second feature to predict the quality score.

With the assistance of domain experts, we extracted a group of common words and grouped them based on their functionality in the description. For the sake of generalization, we defined six different elements for the words. They are: [object, material, property, contain, gender, functionality]. We used binary values to represent whether the description contains the element or not. One example of such a vector is shown in Figure 6.3.

6.4.3 Linguistic Features

In addition to the features mentioned above, we also looked into linguistic features concerning the item description.

We tried Parts of speech(POS), and perplexity features extracted from the GPT2 model [RWA^{+}19]. However, after checking their performance, neither feature was deemed useful. A potential explanation is that most of the descriptions only have three or four words, and contain no grammar structure.

We therefore used the following shallow language features: the length of the description, whether or not a certain word belongs to the English dictionary, and the ratio of English words vs non-English words.

6.4.4 Ordinal Regression

As the target in our training data has an ordinal interpretation, we trained an ordinal regression model for the prediction of the score. The basic idea of the ordinal regression model is to use multiple binary regression models to address the ordinal issue.

Following this principle, a task that has n different classes, can be converted to a binary vector $\{y_0, y_1, ..., y_n\}$. When converting the label $Y = r$ to the vector the following constraint holds:

$$y_i = 1 \, if \; i < r, 0 \, otherwise \tag{6.1}$$

For example, when the number of categories is 5, $Y = 3$ can be converted to the binary vector $[1, 1, 1, 0, 0]$. For each binary value in the vector, we can train a binary regression model.

In our experiments, we use the implementation available in the Mord repository [PBG17]

6.5 Results and Discussion

We perform our evaluation by conducting two analyses. The first analysis aims at evaluating the machine learning model developed by checking how well the model can distinguish different labels. The second analysis consists of a model performance evaluation.

In order to validate whether our proposed model can distinguish the labels or not, we checked the statistical significance for the prediction value distribution on each label. The distribution of prediction value versus the true label plot can be seen in Figure 6.4. One-way ANOVA was used to perform a statistical significance analysis. In our experiment, the p-value is 1.1e-37, which is far less than the common significance level of 0.05. This proves that the predicted values for each label belong to distinguishable levels, the second part of the experiment is then meant to check if these levels are also close to the ground truth provided by the experts.

As for the accuracy, we compared our solution with one of the commonly used sentence embedding method: Universal sentence encoder [CYK$^+$18]. The approaches evaluated were: Our proposed feature, the USE sentence embedding features and the concatenation of both. An ordinal regression model was trained on top of these models. We also compared the distance between the true label and predicted label since the label has an ordinal interpretation. The result is shown in table 6.2.

The results show that our proposed approach has the best performance among the tested feature types. Almost 90% of the data prediction are within 1 label distance. Examining the model based on USE and the one using concatenated feature, it is possible to notice that USE features actually deteriorate the performance. Considering that the data set is relatively small, the model might have difficulties in generalizing when using the USE features.

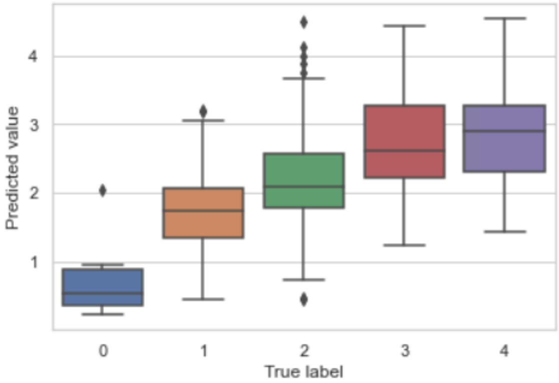


Figure 6.4: Score distribution

Feature	Acc	Dis ≤ 1	Dis ≤ 2	Dis ≤ 3
Our proposal	**45.17**	**89.12**	**99.18**	**100.0**
USE	36.76	84.39	98.15	99.79
Concatenation	42.51	85.01	98.56	99.79

Note: The Distance is defined as: Dis = abs(True label - Predicted label)

Table 6.2: Results

6.6 Conclusion

In this paper, we presented a solution to evaluate the product description quality with a ground truth of 2000 training items and 500 testing items. Considering the small size of the dataset, the proposed model is hybrid and it combines transfer learning, knowledge from a hierarchical classification task, linguistic features, and domain knowledge.

The main result is that the proposed model can assess the quality of a product description closely to human experts. In addition, the proposed features can easily be applied to similar problems such as determining the quality of the customer product review or identifying bots on social media platforms. Potential future work could be to consider a feedback loop at the side of the customer in order to understand if the customer would correct their description of the product, if this is of bad quality. Additionally, different representations of model reliability could be investigated. Last but not least, it would be worth looking into additional linguistic features.

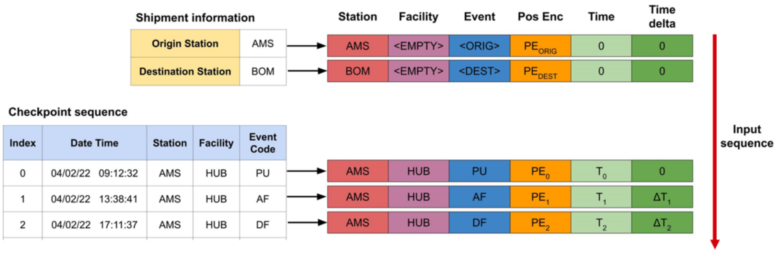
Chapter 7

Language Modeling in Logistics: Customer Calling Prediction

Customer centers in logistics companies deal with many customer calls and requests daily. One of the most common calls is related to requesting an update on the shipment status. Proactively sending message updates to customers can reduce the number of calls. If a machine learning model could predict shipments leading to a customer call based on its journey, it could be possible to proactively send message updates only to customers likely to make a call. Therefore, reducing the workload in the customer center while increasing customer satisfaction. However, naively sending updates to everyone can cause unnecessary anxiety to people who do not want it, thus leading to customer dissatisfaction or even more calls. Therefore, we need to employ selective classification to only proactively update customers when the model is confident. In this paper, we formulate the shipment journey as a variant of a language model. Specifically, we treat checkpoints (station, facility, time, event code) as tokens and predict the next checkpoint(station, facility, time delta, event code). Our core insight is that shipment checkpoints follow a set of rules that dictate the possible sequence of checkpoints. This is similar to how grammar rules dictate which words can follow another. Despite remaining a difficult problem, our experiments show that features learned by modeling shipment checkpoints as a language model can improve customer calling prediction. Moreover, we also show how to use uncertainty analysis on the samples to gain a better understanding of the model performance.

This chapter is based on the following publication:

Xi Chen, Giacomo Anerdi, Daniel Stanley Tan, Stefano Bromuri. *ESANN 2023 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, DOI: 10.14428/esann/2023.ES2023-78, 2023, Language Modeling in Logistics: Customer Calling Prediction.



Origin Station	AMS
Destination Station	BOM

Index	Date Time	Station	Facility	Event Code
0	04/02/22 09:12:32	AMS	HUB	PU
1	04/02/22 13:38:41	AMS	HUB	AF
2	04/02/22 17:11:37	AMS	HUB	DF

Station	Facility	Event	Pos Enc	Time	Time delta
AMS	<EMPTY>	<ORIG>	PE_{ORIG}	0	0
BOM	<EMPTY>	<DEST>	PE_{DEST}	0	0
AMS	HUB	PU	PE_0	T_0	0
AMS	HUB	AF	PE_1	T_1	ΔT_1
AMS	HUB	DF	PE_2	T_2	ΔT_2

Figure 7.1: Data structure

7.1 Introduction

Customer centers of logistics companies are responsible for managing a diverse range of requests from a multitude of customers on a daily basis. The most frequent request is the demand for shipment progress updates. The customer center receives over a million of those calls each month globally, representing a significant workload. Logistic companies could regularly send shipment updates. However, not everyone is keen on receiving them. In fact, unwanted notifications have been shown to have adverse effects such as increased anxiety [ERA$^+$21], which leads to reduced customer satisfaction, and in some cases even more calls. Hence, we need to use selective classification to reject the cases where the model is not confident depends on the business requirements.

By leveraging machine learning algorithms to predict shipments that are likely to prompt customer inquiries, we can proactively send message updates only to customers that are likely to call, thereby reducing the number of calls received. Every percentage of reduction in call volume can result in substantial savings in workload and enhanced customer satisfaction.

From a machine-learning perspective, predicting customer calling based on a shipment's journey can be seen as a time-series classification task with a binary target [TVN$^+$22]. The shipment journey consists of a sequence of checkpoints as Figure 7.1 shows. Each checkpoint contains a timestamp, location, and event code that indicates what happened to a shipment.

Every logistics company has its own process rules that determine the sequence of checkpoints that a shipment makes during its journey. This shipment journey can be treated as a directional graph if the origin and destination are known. The way checkpoints are generated during this journey also follows a certain order.

For example, a shipment should always have an arrival event before a departure event. However, the sequence of the checkpoints is not always fixed; it also depends on the schedule and incidents such as missing flights, transportation delays, etc.

With these observations, we hypothesize that shipment journeys can be formulated as a language modeling problem [WLS20], where checkpoint events taking place concerning the shipment are represented following a set of "grammar" rules dictated by the logistic process.

Moreover, for such a complex time series problem involving a series of checkpoints, stations, and timestamps, understanding the data through a common descriptive approach is challenging. Therefore, we use uncertainty to gain insights into model performance, which indicates that customer call timing exhibits random behavior.

In this paper, we show how to formulate the prediction of shipment journeys by means of a variant of a language model. We demonstrate that customer calling prediction from shipment journeys can benefit from pre-training on a large unlabeled collection of data where the only supervision comes from predicting the next checkpoint, similar to how large language models [TLI$^+$23; BMR$^+$20] have demonstrated that pre-training models can significantly improve downstream tasks [HLM19; PLH$^+$23]. Moreover, we show how uncertainty can be utilized to better understand the data and offer insights for improving model performance. Our findings suggest that modeling shipment journeys as if they were sentences in a language model has the potential to help with various tasks in the logistics domain and that treating checkpoints and journeys as words and sentences is appropriate from a modeling perspective, thus opening many possibilities for future research.

The rest of this paper is structured as follows: Section 7.2 explains the method used in this paper; Section 7.3 shows the details of the experiment; The results and discussion are covered in Section 7.4; Section 7.5 contains the conclusion and the future work.

7.2 Method

7.2.1 Shipment journey as a language model

Similarly to the case of a language model predicting the next word, we train our model to predict the details of the next checkpoint in a shipment journey, as shown in Figure 7.2. This language model based pre-training allows our model to implicitly learn the logistic process and its rules.

We adopt a decoder-only architecture [BMR$^+$20; DXS$^+$23] consisting of six decoder layers. In our experiments, checkpoints are represented by their station information, facility, event code, and time stamp, as shown in Figure 7.1. Station information, facility, and event code are categorical variables that we encode using an embedding layer. For the timestamp, we encode month, day, year, day of week, hours, and minutes using cyclical feature encoding.

Since the time information has a big variance, it is not clear whether it

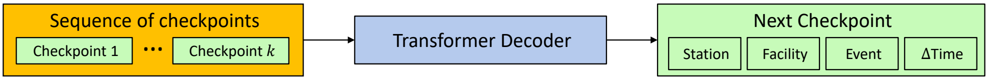


Figure 7.2: Pre-train target.

would be helpful to make it a prediction target. Therefore, we experimented on two different pre-training targets. The first variant of our model only predicts the station, event code, and the facility. This will help the model understand the sequence of events. The second variant includes the transition time, which is the time difference (or time delta) between the previous checkpoint and the next checkpoint.

7.2.2 Customer calling prediction as downstream task

After the language model based pre-training, we fine-tune the model on the target of predicting customer calls, given a shipment journey consisting of a sequence of checkpoints. This is done by replacing the last output layer to output a binary prediction target optimizing a binary cross-entropy loss.

Time duration of a particular checkpoint is crucial information for customer calling predictions. As a matter of facts, if a shipment is stuck at the same checkpoint for a long time, then it is highly likely that the customer will complain. However, simply using the first k checkpoints as input does not indicate how long the shipment has been on the k-th checkpoint. Therefore, we append an end checkpoint at the $k+1$ position with a synthetic time stamp to indicate the duration of the k-th checkpoint and a special 'end' token for facility, station, and event codes.

During training, we augment the negative samples by appending end checkpoints in between two checkpoints where customers did not call. The idea is that if the customer does not call between checkpoint k and checkpoint $k+1$, then an artificial checkpoint $k+1$ with an earlier time stamp will also not trigger a call from the customer. For the positive samples, we generate end checkpoints with the same time stamp as the beginning of the call from the customer.

Due to the nature of the calls, the number of customers who call is significantly less than the number of customers who do not call. In our case, the ratio between positive and negative samples is approximately $1:19$. Thus, we sample the negative ones in such a way that the number of positive and negative samples is balanced. Specifically, we under-sample the negative samples in each epoch training.

7.3 Experiment

7.3.1 Data

The data that we used in this experiment comprises six months of shipments toward one country. For each shipment, we have corresponding checkpoint sequences as shown in Fig. 7.1.

Overall the data set contains a sample of 2.49 million shipments, where in 5.2% of the cases the customer called to obtain more information. In order to give some insights into the difficulty of this problem, Table 7.1 below shows the proportion of customer calls on a particular shipment event. The data is not only highly imbalanced, but also contains various types of noise and uncertainties. For example, some customers call at random times purely out of concern. Moreover, even for the shipments that present the same status, some customers call while others do not because of external factors, such as their personal situations or urgency of receiving the shipment.

We use three-month data for training, and half a month for validation and testing respectively. In order to evaluate statistical significance, we apply a five-fold rolling cross-validation with a window size of half a month in this experiment.

7.3.2 Implementation details

As for the transformer decoder, we use an Adam optimizer with a learning rate of $1e-5$. The number of heads is 6, and the dimension of the model is 512, with 6 layers of the decoder.

7.4 Results & Discussion

In this experiment, we evaluated three different models. The first model is a transformer model that is directly trained on the target. This acts as the standard classification baseline wherein we do not perform any language model based pre-training. The second and third models are the fine-tuned models with language model based pre-training. Specifically, the second model is pre-trained without a time delta, while the third model includes a time delta as a pre-training target.

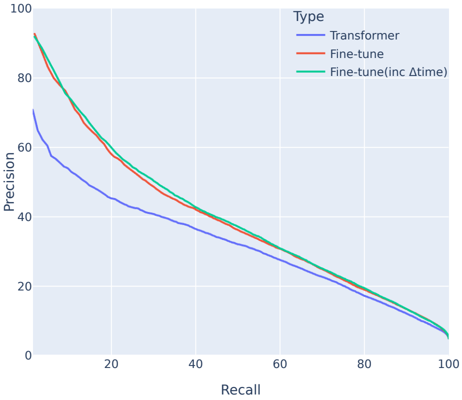


Figure 7.3: The precision-recall plot.

Table 7.1: The proportions of the last event before the customer called with the non-call shipments for the same event. We show only the top 4 events related to customer calls.

Event code	A	B	C	D
Call	8.1%	5.2%	4.3%	3.2%
Non-call	91.9%	94.8%	95.7%	96.8%

The results are shown in the Table 7.2. We report the average recall from the five-fold cross-validation at different precision values. A complete precision-recall curve is visualized in Figure 7.3. Based on the results, we can observe that pre-training the transformer model on the shipment journey can improve the performance of customer call prediction. The difference is statistically significant when compared to the transformer without pre-training. We can also observe that the performance of the fine-tuning with time delta is marginally better than without. Time delta is important information when it comes to understanding the shipment journey. Predicting the time delta can help the model to understand the intrinsic shipping logic and the average duration of each checkpoint, thus leading to better performance when fine-tuning the downstream tasks.

The recall values are not particularly high on all models, which is reflective of how difficult the task is. The highly imbalanced data coupled with the randomness of customer call behavior makes it challenging to predict customer calls accurately. However, we would like to note that the level of performance of the models is already useful for industry usage. Depending on the business needs, we can always trade off precision & recall by moving the threshold (Figure 7.3). From a business perspective, based on 1 million calls per month, if we use a precision threshold at 50% and send the customer an update message, even if only half of the customers can be prevented from contacting the logistic company, this can reduce 15% of the calls, which accounts for 150k calls per month.

Due to the high complexity of the data, understanding it through common descriptive analysis is rather challenging. We utilize the uncertainty method described in Mukhoti et al. [MKvA$^+$23] to help better understand the model performance. As Figure 7.4 shows, the distributions of epistemic uncertainty

Precision	Random Guess	Transformer	Fine-tune	Fine-time (w/ time delta)
40	n.a	32.95	43.50 (2.9e-3)	**45.55** (3.6e-3)
50	5.2	14.05	29.29 (6.0e-3)	**30.86** (9.3e-4)
60	n.a	4.80	19.39 (1.7e-3)	**20.16** (2.5e-3)
70	n.a	1.58	12.04 (5.2e-3)	**13.14** (8.3e-3)
80	n.a	0.76	6.60 (7.6e-3)	**7.65** (9.6e-3)

Table 7.2: Recall under the different precision threshold. The number in the bracket is the P-value from paired student T-test compared with Transformer. A precision threshold can to set in order to decide which customers should be updated.

for both correctly and incorrectly classified samples overlap significantly. This overlap suggests that the model behaves poorly in these areas, and also indicates that there might be randomness when a customer decides to call. Checking those data with similar uncertainty levels reveals that shipments with the same patterns vary in customer response; some receive customer calls while others do not. It gives insight that integrate customer behavior information might improve the performance further.

7.5 Conclusion & Future work

In this chapter, we showed that it is possible to formulate the shipment journey as a variant of the language model. This opens the possibility of implementing a large language model that can be used in the logistic domain. The downstream task customer calling prediction can also be a benefit for the logistic company.

Future work concerning customer contact prediction could imply looking into data cleansing and uncertainty measurement approaches [HU20; HFB19] as there is certain randomness involved in the data. For the language model itself, further analysis could be performed on fine-tuning it towards various downstream tasks to validate its usage in other logistic case studies.

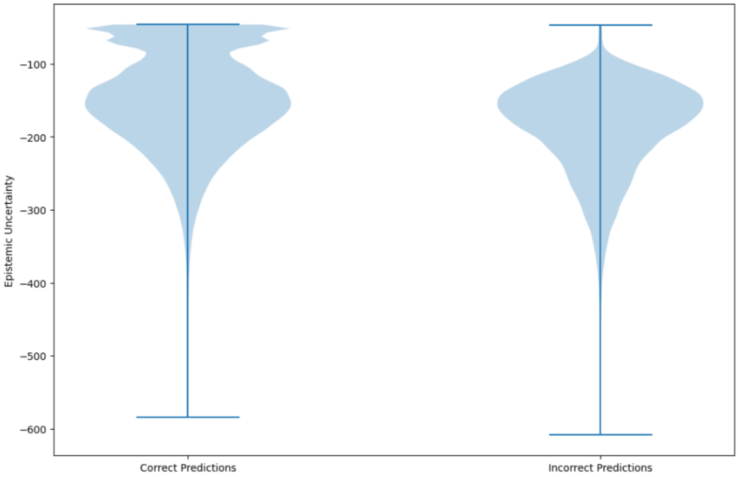


Figure 7.4: A violin plot is used to visualize the distribution of epistemic uncertainty in "Correct Predictions" and "Incorrect Predictions." The width of the violin at different levels of the vertical axis represents the density of the data, with wider sections indicating a higher density of values at that range. In theory, misclassified samples (right side) tend to have higher epistemic uncertainty (e.g., outliers, novelty, etc.) compared to correctly classified samples (left side). However, the distribution in the higher uncertainty region shows a similar density for both categories. This suggests that the model struggles to make decisions in those areas. Combined with sample checks, this indicates the randomness in customer behavior when deciding whether to call customer service or not.

Chapter 8

Conclusions

This thesis is about selective classification and its usage in the logistic domain. It sets forth the following objectives:

1. To demonstrate the usage of selective classification in the logistic domain.
2. To investigate improved confidence score that are effective for both public datasets and HS Codes classification issues.
3. To explore how confidence score can be utilized to better understand data quality.

This chapter summarizes the thesis, addresses each research question concerning limitations and future work, and discusses the considerations of selective classification in terms of deployment and its relation to Large Language Models (LLMs) within the context of this thesis. Finally, it concludes by discussing the social impact of our findings.

8.1 Research Question

This thesis aims to answer the following three research questions:

- What deep learning models are suitable for hierarchical text classification (HTC)?
- How to use the error pattern to design a better confidence score?
- How to use the confidence score to analyze data quality (1. Whether the data contains enough information for the category. 2. Model uncertainty analysis to understand data label noise)?

Each question has been investigated in the thesis chapters and is further discussed in this section.

8.1.1 What deep learning models are suitable for hierarchical text classification (HTC)?

To answer this question, we investigated existing HTC architectures in Chapters 3 and 5, and decided to focus on hierarchical global classification. The Seq2seq model is a common choice for hierarchical global classification due to its architecture and strong performance. We compared traditional logistic regression, LSTM-based seq2seq, and transformer-based seq2seq models for HTC. Our findings establish that the transformer-based seq2seq model is highly effective for HTC problems.

This finding is significant as it directs logistics companies towards adopting this model for improved classification accuracy and efficiency in HSCode classification, leading to potential cost savings and enhanced operational workflows.

Limitation

The research is not guaranteed to find the best performance model on HTC problem as we did not explore the variants of the transformer-based seq2seq models. Additionally, we noticed that the lack of public datasets for the HTC problem made it challenging to conduct a fair comparison.

Future work

One of the limitations of the seq2seq model is that it can generate invalid/non-existing labels. Those invalid labels can be filtered out by checking against the valid labels. So future work can extend the beam search so that the invalid labels will get removed during the generating process. Moreover, considering the strong knowledge of the existing LLMs, fine-tuning LLMs for HTC problems might also be worth trying.

8.1.2 How to use the error pattern to design a better confidence score?

The primary challenge for a better confidence score lies in 1) identifying a target that enables sample ranking based on their likelihood of misclassification or ease of classification, and 2) ensuring the model's ability to generalize from this target. Generalization poses a significant issue as there is no assurance of the model's capacity to generalize the target effectively. In this work, the error pattern we are looking at is the error of the classifier, the error of rejection in both True class probability (TCP) and Failure detection (FD).

Chapters 4 and 5 address this question. To evaluate the generalizability TCP rejection methodology, we applied it to the HS Codes and importer prediction problems. We found that the TCP and FD approach struggles to generalize misclassified samples in HTC problem. By analyzing the error where the model and rejection happened, we introduced a new rejection criterion, the "confidence range", which leverages model failures and TCP. This proposal can further enhance performance.

Limitation

Due to the nature of these confidence scores, it is difficult for us to determine whether the model's failure to generalize well is due to the inherent nature of the data or a poorly designed confidence score. This makes post-analysis of the results quite challenging. This could also be the future work: utilizing uncertainty to better understand the confidence range and identify areas for improvement.

Future work

Our contribution integrates model failure with TCP to introduce a 'range target', enhancing the model's generalization capabilities in predicting the confidence score. Future work could involve incorporating regularization strategies to prevent feature collapse, as suggested by [vASJ$^+$21]. Moreover, we could leverage uncertainty as an additional feature to predict confidence scores.

8.1.3 How to use the confidence score to analyze data quality?

This research extends into evaluating data quality through confidence score as described in Chapters 6 and 7. The confidence score is a good indicator of whether the model can classify a sample correctly, and thus, a good indicator of sample quality. We used confidence scores derived from different methodologies to identify data quality issues.

In Chapter 6, we proposed using the confidence score as a transfer learning feature, combined with other domain and linguistic features, to build an ordinal classifier to identify data quality. This approach is rather useful especially when it is unrealistic to get a large amount of the labeled data to apply an end-to-end model.

Additionally, distinguishing whether errors originate from the data or the model provides crucial insights into the limitations faced when dealing with complex data problems. This distinction has practical implications for the entire machine learning pipeline, as demonstrated in Chapter 7. Analyzing the uncertainty of samples offers valuable indications of whether errors stem from the data itself or the modeling process. Thus, provides the direction to enhance the model performance.

Limitation

Due to a lack of other domain data and knowledge, all the data quality-related work detailed in this thesis was applied only to logistics domain-related problems. Therefore, it would be better if those methodologies could be tested in other domains to validate their generalizability.

Future work

Future work could focus on developing a framework that automatically performs both classification and highlights data issues for industrial applications. Also, in this work, we did not further develop uncertainty-based methodologies. Instead, we applied the methodology from [MKvA$^+$23] to gain a better understanding of predictions from both data and model perspectives. Modeling uncertainty itself is a challenging task. Quantifying uncertainty in classification tasks is undoubtedly beneficial. The most obvious difference between aleatoric and epistemic uncertainty is that one is reducible and the other is not. However, separating these two types of uncertainty is difficult. There is a thin line between them, and they can even be convertible depending on our modeling choices [DD09; KG17]. Therefore, improving our ability to distinguish between these two types of uncertainty will remain a challenging task.

Regarding rejection: Most current methodologies use either aleatoric or epistemic uncertainty for rejection. Exploring a better combination of these two types of uncertainty as rejection criteria, as suggested by Vazhentsev in [VKT$^+$23], could be one of the directions for future research.

8.2 Considerations about Industrial deployment

When selecting a rejection method for industry usage, its applicability must be carefully considered. This is particularly important in scenarios where the balance between model performance and scalability is delicate, as decisions must weigh the cost of deployment against the benefits. Despite their theoretical robustness, Bayesian-based methodologies have seen limited popularity in certain industry applications due to their high computational cost.

Our early work in HSCode classification problem consciously bypassed certain methodologies due to concerns over their applicability in a real-world industry setting. Two critical factors need to be considered when choosing the deployment of these methods in the industry: computational cost and transferability. Also, model performance monitoring is important after deployment.

Computational cost. In an industrial environment, where time and resources are often limited, the computational cost becomes a significant barrier to the adoption of certain methodologies. Especially since nowadays everything is cloud-based, computational power is directly associated with a cost. Therefore, we always need to trade off between performance and usage of cloud infrastructure.

Techniques that can save computational costs, such as caching and model compression, are commonly used in industry. Caching helps by storing frequently accessed data, thus reducing the need for repeated data retrieval operations. Model compression can reduce the size and complexity of machine learning models, making them more efficient to run while maintaining most of their performance.

In the work of Chapter 4, instead of using a classifier to predict everything, we used simple matching logic to predict the easy samples, saving time and computational cost. Additionally, caching was applied to the HS Codes prediction problem by storing high-frequency descriptions in a vector database. For each new sample, we searched the vector database first before classifying it.

We also took this into account when choosing the rejection approach. Rejection approaches with higher computational power (e.g., KNN, ensembles, etc.) were generally not our first choice. Overall, a single-pass neural network is more appealing to the industry.

Transferability. The ability to apply a method across different models with minimal adjustment is another crucial consideration. A method that works well in an academic setting may not be attractive to the industry if it requires a lot of customization. This could be because the model used in the industry is not easily adaptable to the solution or because there is a lack of capability to make such changes to the model. Thus, the softmax baseline is still the first choice for the industry as long as the performance is acceptable.

Recent developments in uncertainty-based methodologies have introduced certain requirements for model adaptation, notably modifications to the model's architecture to incorporate hidden layers and specific normalization techniques. These adjustments are essential to ensure that the hidden layers produce meaningful outputs. Such modifications, however, complicate the transferability of these approaches to other models due to the specialized nature of the changes required. In contrast, ranking-based approaches offer a more adaptable solution for industry applications, primarily due to their simpler integration process and good performance. That is also why we decided to develop those methods further.

Monitoring. Machine learning solutions generally require constant monitoring of performance to avoid data drift, a phenomenon where data patterns change over time. This process is crucial for selective classifiers as well. In the context of a logistics company, such as DHL, this becomes even more critical for the following reasons: 1. The package that the customer shipped has seasonal and yearly shipping patterns. 2. The types of products being shipped can change over time due to varying global activities. Therefore, continuous performance monitoring is essential to mitigate risks and ensure the models remain accurate and reliable.

8.3 Considerations about LLMs

In recent years, the advent of Large Language Models (LLMs) has marked a transformative period in the machine learning community, demonstrating successful applications across various domains [NMH$^+$24; CAT$^+$; HBL$^+$23].

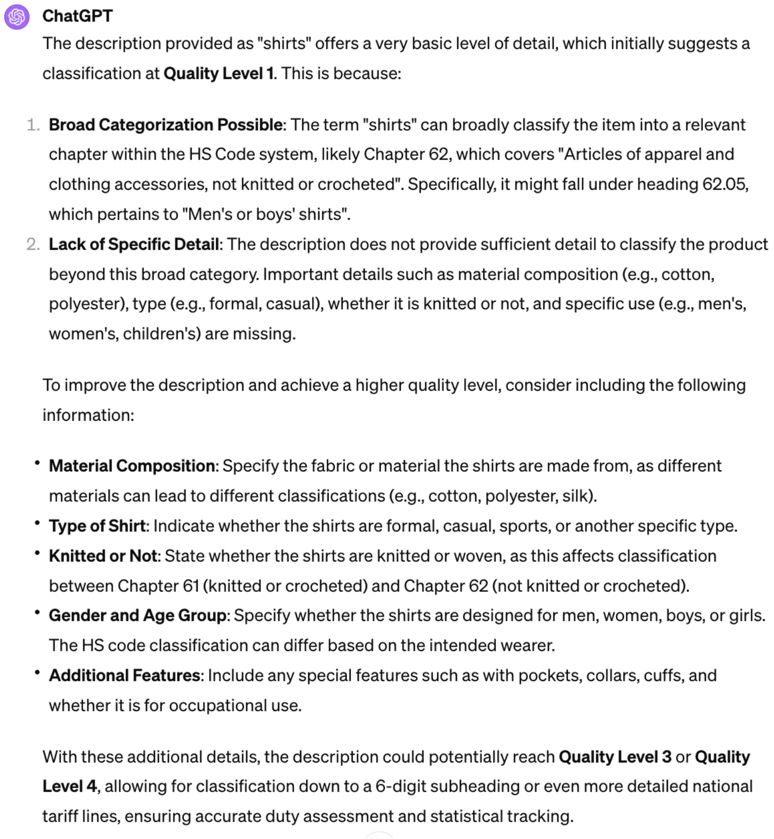
LLMs that have been trained on enormous datasets can evaluate data quality through few-shot learning techniques, as detailed in [ESA$^+$23]. We tried a quick exploration of LLM on the problem that was tackled in Chapter 6 and 3. ChatGPT-4 exhibited remarkable classification accuracy and reasoning capabilities to determine the shipment description quality, as shown in Figure 8.1.

This leads us to also consider the potential integration of LLMs within the framework of selective classification. Given their strong capability to understand the quality of shipment descriptions and classify them into corresponding HS codes. Fine-tuning the LLMs on the HS Code classification training dataset will enable LLMs to simultaneously predict rejection options and the appropriate class. Such an approach could revolutionize selective classification, leveraging the inherent strengths of LLMs to enhance decision-making processes in machine learning applications.

The broad usage of Large Language Models (LLMs) also poses some ethical and social risks [WMR$^+$21]. Even when LLMs are not performing classification tasks, the capability to reject misuse requests is crucial [PPC$^+$23]. Ensuring that LLMs can effectively reject inappropriate inputs and/or outputs will enhance their reliability and safety in industrial applications. This is an interesting research direction [MLG23; YKG$^+$24] for selective classification and will become increasingly important as the power of LLMs grows.

8.4 Societal Impact

The thesis not only focuses on methodology but also demonstrates the capability of selective classification in real-world use cases within the logistics domain. Here, four aspects will be discussed: Economic Impact, Business understanding, Process and Reliability, and Broader Societal Implications.



ChatGPT

The description provided as "shirts" offers a very basic level of detail, which initially suggests a classification at **Quality Level 1**. This is because:

1. **Broad Categorization Possible**: The term "shirts" can broadly classify the item into a relevant chapter within the HS Code system, likely Chapter 62, which covers "Articles of apparel and clothing accessories, not knitted or crocheted". Specifically, it might fall under heading 62.05, which pertains to "Men's or boys' shirts".
2. **Lack of Specific Detail**: The description does not provide sufficient detail to classify the product beyond this broad category. Important details such as material composition (e.g., cotton, polyester), type (e.g., formal, casual), whether it is knitted or not, and specific use (e.g., men's, women's, children's) are missing.

To improve the description and achieve a higher quality level, consider including the following information:

- **Material Composition**: Specify the fabric or material the shirts are made from, as different materials can lead to different classifications (e.g., cotton, polyester, silk).
- **Type of Shirt**: Indicate whether the shirts are formal, casual, sports, or another specific type.
- **Knitted or Not**: State whether the shirts are knitted or woven, as this affects classification between Chapter 61 (knitted or crocheted) and Chapter 62 (not knitted or crocheted).
- **Gender and Age Group**: Specify whether the shirts are designed for men, women, boys, or girls. The HS code classification can differ based on the intended wearer.
- **Additional Features**: Include any special features such as with pockets, collars, cuffs, and whether it is for occupational use.

With these additional details, the description could potentially reach **Quality Level 3** or **Quality Level 4**, allowing for classification down to a 6-digit subheading or even more detailed national tariff lines, ensuring accurate duty assessment and statistical tracking.

Figure 8.1: Example of ChatGPT-4 in classifying description quality

Economic Impact. This thesis showcases the application of selective classification in addressing HSCode classification, importer prediction, and customer calling prediction problems within the logistics sector. The research provides evidence that the application of selective classification can lead to multi-million euro savings. By automating operational processes, logistics companies can partially streamline their business operations and allocate resources more efficiently, resulting in substantial cost reductions. This economic advantage not only enhances the profitability of individual companies but also contributes to the overall economy by improving the competitiveness of the logistics sector.

Selective classification offers a higher tolerance concerning data quality since it has the capability of rejecting ‘not known or not confident’ cases. This increases the likelihood of success compared to a fully automated solution that requires better processes or setups.

Business understanding. Analyzing rejected samples from selective classification can enhance our business understanding and uncover hidden issues. Business understanding is a crucial process in data science projects, as it provides the primary information source to validate whether the data accurately reflects reality.

Generally, we assume that domain experts can address most business-related questions. However, in this research, we observed that this is not always the case in the logistics domain. Shipment data undergoes a lengthy process from pickup to delivery. In each process, the data might be modified. Therefore, it is challenging to find a single individual who comprehends the entire process. Most domain experts are only familiar with the specific processes they work on, making it difficult to obtain a complete picture of everything that has happened to the data.

Throughout the research, we found that rejected samples are often associated with outliers, inconsistent data, or new products. Analyzing those samples allows us to validate business processes and uncover hidden issues.

Process and Reliability. Data quality is always an issue in industry. Many machine learning projects fail due to insufficient data quality [RA20]. A traditional classification solution suffers from “garbage in, garbage out” problems. This might be caused by excessive noise in the data, flaws in the data collection process, or system errors. To fully automate such a solution one would have to fix data quality issues first.

Selective classification can still automate part of the classification processes even with poor data quality, thus yielding benefits directly instead of convincing businesses to improve the data quality first. This makes the selective classification more appealing when it comes to industry usage, especially considering

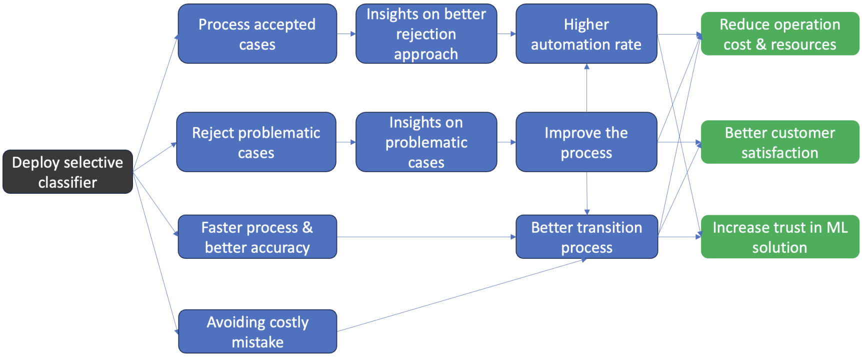

Deploy selective classifier
Process accepted cases
Reject problematic cases
Faster process & better accuracy
Avoiding costly mistake
Insights on better rejection approach
Insights on problematic cases
Higher automation rate
Improve the process
Better transition process
Reduce operation cost & resources
Better customer satisfaction
Increase trust in ML solution

Figure 8.2: Pathway to impact

that improving the data quality takes a lot of time and effort.

The application of selective classification in logistics not only improves efficiency and sustainability, but also enhances reliability of the processes. Figure 8.2 illustrates how a selective solution can be sold to a company.

In logistics, particularly in clearance processes where errors can cause multi-day delays, accurate classification and high data quality are crucial for reducing risks, improving operations, and increasing customer satisfaction. After deploying the selective classifier, samples where the model can achieve the desired precision will be automated, streamlining the process. This will lead to a smoother process. The remaining samples will be reviewed by the agent. Analyzing the rejected samples will provide a better understanding of the process, exposing issues and offering insights for improved rejection methods. Since the benefit of each percentage of automation can be visualized, it gives more motivation to fix the data quality issue and investigate a better rejection methodology. Naturally, this leads to a virtuous cycle that constantly improves processes, customer satisfaction, and creates buy-in for machine learning solutions.

Broader Societal Implications. Beyond the direct impacts on the economy, this research also has a broader impact on society, as all the work presented is directly applied in industry setups. Chapters 3, 4, and 7 demonstrate how selective classification can be adopted in industry settings. These examples provide valuable insights for other sectors on implementing selective classification in their domains. This is particularly relevant in fields like biomedicine, where precise diagnostics are essential for patient care [GJH+20], and in manufacturing, where automated quality control is vital for maintaining production standards [ZDM20].

Additionally, our problem-solving approach can be replicated in other areas, which is particularly valuable for companies just beginning to explore machine-learning solutions. This method provides a quicker way to demonstrate benefits and naturally leads to higher success rates for projects.

Bibliography

References

[Abd15] Aida Krichene Abdelmoula. Bank credit risk analysis with k-nearest-neighbor classifier: case of tunisian banks. *Accounting and Management Information Systems*, 14(1), 2015.

[ABP+22] Taiga Abe, Estefany Kelly Buchanan, Geoff Pleiss, Richard Zemel, and John P Cunningham. Deep ensembles work, but are they necessary? *Advances in Neural Information Processing Systems*, 35, 2022.

[AP19] Ion Madrazo Azpiazu and Maria Soledad Pera. Multiattentive recurrent neural network architecture for multilingual readability assessment. *Transactions of the Association for Computational Linguistics*, 7, 2019.

[APH+21] Moloud Abdar et al. A review of uncertainty quantification in deep learning: techniques, applications and challenges. *Information fusion*, 76, 2021.

[ASS+20] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. *Advances in Neural Information Processing Systems*, 33, 2020.

[Ati05] Amir F Atiya. Estimating the posterior probabilities using the k-nearest neighbor rule. *Neural computation*, 17(3), 2005.

[AYR16] Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. Automatic text scoring using neural networks. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2016.

[Bar12] David Barber. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[BB21] Alexander Brinkmann and Christian Bizer. Improving hierarchical product classification using domain-specific language modelling. *IEEE Data Eng. Bull.*, 44(2), 2021.

[BCB15] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[BDD+16] Mariusz Bojarski et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[BFF+04] John Blatz et al. Confidence estimation for machine translation. In *Coling 2004: Proceedings of the 20th International Conference on Computational linguistics*, 2004.

[BGR+99] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is “nearest neighbor” meaningful? In *International Conference on Database Theory*. Springer, 1999.

[BMG17] John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani. Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks. *arXiv preprint arXiv:1707.02476*, 2017.

[BMR+20] Tom Brown et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 2020.

[Böh92] Dankmar Böhning. Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1), 1992.

[BW08] Peter L Bartlett and Marten H Wegkamp. Classification with a reject option using a hinge loss. *Journal of Machine Learning Research*, 9(8), 2008.

[CAT+] Xi Chen, Giacomo Anerdi, Daniel Stanley Tan, and Stefano Bromuri. Language modeling in logistics: customer calling prediction.

[CAT+23] Xi Chen, Giacomo Anerdi, Daniel Stanley Tan, and Stefano Bromuri. Language modeling in logistics: customer calling prediction. In *Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium*, 2023.

[CBT22] Xi Chen, Stefano Bromuri, and Daniel Stanley Tan. Confidence range: bridging failure detection and true class probability on selective hierarchical text classification. *Available at SSRN 4244490*, 2022.

[CBV21] Xi Chen, Stefano Bromuri, and Marko Van Eekelen. Neural machine translation for harmonized system codes prediction. In *International Conference on Machine Learning Technologies*, 2021.

[CBV22] Xi Chen, Stefano Bromuri, and Marko Van Eekelen. A transfer learning approach to predict shipment description quality. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022.

[CDT+95] Luigi Pietro Cordella, Claudio De Stefano, Francesco Tortorella, and Mario Vento. A method for improving classification reliability of multilayer perceptrons. *IEEE Transactions on Neural Networks*, 6(5), 1995.

[CGK17] Miriam Cha, Youngjune Gwon, and HT Kung. Language modeling by clustering with word embeddings for text readability assessment. In *ACM on Conference on Information and Knowledge Management*, 2017.

[CGZ06] Nicolo Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Hierarchical classification: combining bayes with svm. In *Proceedings of the 23rd International Conference on Machine Learning*, 2006.

[CH04] Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *Proceedings of the 13th ACM international conference on Information and knowledge management*, 2004.

[CH17] Yingjun Chen and Yongtao Hao. A feature weighted support vector machine and k-nearest neighbor algorithm for stock market indices prediction. *Expert Systems with Applications*, 80, 2017.

[Cha20] Francisco Charte. A comprehensive and didactic review on multilabel learning software tools. *IEEE Access*, 8, 2020.

[CHL+21] Qu Cui et al. Directqe: direct pretraining for machine translation quality estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35 of number 14, 2021.

[Cho57] Chi-Keung Chow. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, (4), 1957.

[Cho70] C Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1), 1970.

[CKK+18] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jung-Tae Lee. Cfgan: a generic collaborative filtering framework based on generative adversarial networks. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018.

[CLB+21] Leiyu Chen, Shaobo Li, Qiang Bai, Jing Yang, Sanlong Jiang, and Yanming Miao. Review of image classification algorithms based on convolutional neural networks. *Remote Sensing*, 13(22), 2021.

[CMP+21] Sneha Chaudhari, Varun Mithal, Gungor Polatkan, and Rohan Ramanath. An attentive survey of attention models. *ACM Transactions on Intelligent Systems and Technology*, 12(5), 2021.

[CSD+17] Scott A Crossley, Stephen Skalicky, Mihai Dascalu, Danielle S McNamara, and Kristopher Kyle. Predicting text comprehension, processing, and familiarity in adult readers: new approaches to readability formulas. *Discourse Processes*, 54(5-6), 2017.

[CTB+19] Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. *Advances in Neural Information Processing Systems*, 32, 2019.

[CTG+23] Xi Chen, Daniel Stanley Tan, Prakash Gupta, and Stefano Bromuri. Pair-wise selective classification with dynamic sampling for shipment importer prediction. In *Proceedings of the 2023 15th International Conference on Machine Learning and Computing*, 2023.

[CYK+18] Daniel Cer et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[CZG20] Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: uncertainty estimation without ood samples via density-based pseudo-counts. *Advances in Neural Information Processing Systems*, 33, 2020.

[CZS19] Dallas Card, Michael Zhang, and Noah A Smith. Deep weighted averaging classifiers. In *Proceedings of the conference on fairness, accountability, and transparency*, 2019.

[Dal09] Christoph Dalitz. Reject options and confidence measures for knn classifiers. *Schriftenreihe des Fachbereichs Elektrotechnik und Informatik Hochschule Niederrhein*, 8, 2009.

[DD09] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2), 2009.

[DF21] Francesco D'Angelo and Vincent Fortuin. Repulsive deep ensembles are bayesian. *Advances in Neural Information Processing Systems*, 34, 2021.

[DFC15] Liya Ding, ZhenZhen Fan, and DongLiang Chen. Auto-categorization of hs code using background net approach. *Procedia Computer Science*, 60, 2015.

[DG06] Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, 2006.

[DJS20] Tovly Deutsch, Masoud Jasbi, and Stuart M Shieber. Linguistic features for readability assessment. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, 2020.

[DK82] Alice Davison and Robert N Kantor. On the failure of readability formulas to define readable texts: a case study from adaptations. *Reading Research Quarterly*, 1982.

[DQH+19] Jingtao Ding, Yuhan Quan, Xiangnan He, Yong Li, and Depeng Jin. Reinforced negative sampling for recommendation with exposure data. In *IJCAI*, 2019.

[DQY+20] Jingtao Ding, Yuhan Quan, Quanming Yao, Yong Li, and Depeng Jin. Simplify and robustify negative sampling for implicit collaborative filtering. *Advances in Neural Information Processing Systems*, 33, 2020.

[DSV00] Claudio De Stefano, Carlo Sansone, and Mario Vento. To reject or not to reject: that is the question-an answer in case of neural classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30(1), 2000.

[DT18] Terrance DeVries and Graham W Taylor. Learning confidence for out-of-distribution detection in neural networks. *stat*, 1050, 2018.

[DXS+23] Danny Driess et al. Palm-e: an embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[DYW+19] Li Dong et al. Unified language model pre-training for natural language understanding and generation. In *Advances in Neural Information Processing Systems*, 2019.

[DZY17] Fei Dong, Yue Zhang, and Jie Yang. Attention-based recurrent convolutional neural network for essay scoring. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, 2017.

[El-+10] Ran El-Yaniv et al. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(5), 2010.

[ERA+21] Jon D Elhai, Dmitri Rozgonjuk, Ahmad M Alghraibeh, and Haibo Yang. Disrupted daily activities from interruptive smartphone notifications: relations with depression and anxiety severity and the mediating role of boredom proneness. *Social Science Computer Review*, 39(1), 2021.

[ESA+23] Amine Elhafsi, Rohan Sinha, Christopher Agia, Edward Schmerling, Issa AD Nesnas, and Marco Pavone. Semantic anomaly detection with large language models. *Autonomous Robots*, 47(8), 2023.

[FHL19] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep ensembles: a loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

[FJH+10] Lijun Feng, Martin Jansche, Matt Huenerfauth, and Noémie Elhadad. A comparison of features for automatic readability assessment. In *Coling 2010: Posters*, 2010.

[FR02] Giorgio Fumera and Fabio Roli. Support vector machines with embedded reject option. In *International Workshop on Support Vector Machines*. Springer, 2002.

[FYB18] Youmna Farag, Helen Yannakoudakis, and Ted Briscoe. Neural automated essay scoring and coherence modeling for adversarially crafted input. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018.

[GAG+17] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International conference on machine learning*. PMLR, 2017.

[Gal16] Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 1(3), 2016.

[Gao20] Dehong Gao. Deep hierarchical classification for category prediction in e-commerce system. In *Workshop on e-Commerce and NLP*, 2020.

[Gar83] Kathleen Garland. An experiment in automatic hierarchical document classification. *Information Processing & Management*, 19(3), 1983.

[GAS05] Michael Gamon, Anthony Aue, and Martine Smets. Sentence-level mt evaluation without reference translations: beyond language modeling. In *Proceedings of the 10th EAMT Conference: Practical applications of machine translation*, 2005.

[GE17] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

[GE19] Yonatan Geifman and Ran El-Yaniv. Selectivenet: a deep neural network with an integrated reject option. In *International Conference on Machine Learning*. PMLR, 2019.

[GG15] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

[GG16] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *International Conference on Machine Learning*. PMLR, 2016.

[GHV14] Andrew Gelman, Jessica Hwang, and Aki Vehtari. Understanding predictive information criteria for bayesian models. *Statistics and Computing*, 24(6), 2014.

[GJH+20] Jeremy Goecks, Vahid Jalili, Laura M Heiser, and Joe W Gray. How machine learning will transform biomedicine. *Cell*, 181(1), 2020.

[GPS+17] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*. PMLR, 2017.

[GRG+21] Federica Granese, Marco Romanelli, Daniele Gorla, Catuscia Palamidessi, and Pablo Piantanida. Doctor: a simple method for detecting misclassification errors. *Advances in Neural Information Processing Systems*, 34, 2021.

[GRK+08] Yves Grandvalet, Alain Rakotomamonjy, Joseph Keshet, and Stéphane Canu. Support vector machines with a reject option. *Advances in neural information processing systems*, 21, 2008.

[GRP+23] Eduardo Dadalto Câmara Gomes, Marco Romanelli, Georg Pichler, and Pablo Piantanida. A data-driven measure of relative uncertainty for misclassification detection. In *The Twelfth International Conference on Learning Representations*, 2023.

[GTA+23] Jakob Gawlikowski et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, 2023.

[HBL+23] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. Tabllm: few-shot classification of tabular data with large language models. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023.

[HCL+19] Wei Huang et al. Hierarchical multi-label text classification: an attention-based recurrent network approach. In *ACM International Conference on Information and Knowledge Management*, 2019.

[HDV18] Simon Hecker, Dengxin Dai, and Luc Van Gool. Failure prediction for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018.

[Hel70] Martin E Hellman. The nearest neighbor classification rule with a reject option. *IEEE Transactions on Systems Science and Cybernetics*, 6(3), 1970.

[HFB19] Reihaneh H Hariri, Erik M Fredericks, and Kate M Bowers. Uncertainty in big data analytics: survey, opportunities, and challenges. *Journal of Big Data*, 6(1), 2019.

[HG16] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2016.

[HH14] Michael Alexander Kirkwood Halliday and Ruqaiya Hasan. *Cohesion in english*, number 9. Routledge, 2014.

[HL20] S Hernández and Juan L López. Uncertainty quantification for plant disease detection using bayesian deep learning. *Applied Soft Computing*, 96, 2020.

[HLG+20] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2020.

[HLM19] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*. PMLR, 2019.

[HLY+22] Zongbo Han et al. Umix: improving importance weighting for subpopulation shift via uncertainty-aware mixup. *Advances in Neural Information Processing Systems*, 35, 2022.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8), 1997.

[HU20] Öncü Hazır and Gündüz Ulusoy. A classification and review of approaches and methods for modeling uncertainty in projects. *International Journal of Production Economics*, 223, 2020.

[HW21] Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110, 2021.

[HWC+20] Wu-Chih Hu, Hsin-Te Wu, Hsin-Hung Cho, and Fan-Hsun Tseng. Optimal route planning system for logistics vehicles based on artificial intelligence. *Journal of Internet Technology*, 21(3), 2020.

[HXZ+21] Lijuan Huang, Guojie Xie, Wende Zhao, Yan Gu, and Yi Huang. Regional logistics demand forecasting: a bp neural network approach. *Complex & Intelligent Systems*, 2021.

[HZR+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

[HZW+16] Shuo Hu, Yaqing Zuo, Lizhe Wang, and Peng Liu. A review about building hidden layer methods of deep learning. *Journal of Advances in Information Technology*, 7(1), 2016.

[IPG+18] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In *34th Conference on Uncertainty in Artificial Intelligence 2018, UAI 2018*. Association For Uncertainty in Artificial Intelligence (AUAI), 2018.

[JKG+18] Heinrich Jiang, Been Kim, Melody Guan, and Maya Gupta. To trust or not to trust a classifier. *Advances in neural information processing systems*, 31, 2018.

[JLL+19] Xie Jun, Yu Lu, Zhu Lei, and Duan Guolun. Conditional entropy based classifier chains for multi-label classification. *Neurocomputing*, 335, 2019.

[JOK+12] Xiaoqian Jiang, Melanie Osl, Jihoon Kim, and Lucila Ohno-Machado. Calibrating predictive model estimates to support personalized medicine. *Journal of the American Medical Informatics Association*, 19(2), 2012.

[KB19] Mahmut Kaya and Hasan Şakir Bilge. Deep metric learning: a survey. *Symmetry*, 11(9), 2019.

[KBH+17] Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. Hdltex: hierarchical deep learning for text classification. In *International Conference on Machine Learning and Applications*, 2017.

[KG17] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017.

[KL16] Hyun Kim and Jong-Hyeok Lee. A recurrent neural networks approach for estimating the quality of machine translation output. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016.

[KLM19] Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. *Advances in Neural Information Processing Systems*, 32, 2019.

[KLN17] Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*, 2017.

[KLY97] Ron Kohavi, Pat Langley, and Yeogirl Yun. The utility of feature weighting in nearest-neighbor algorithms. In *Proceedings of the Ninth European Conference on Machine Learning*. Citeseer, 1997.

[KMF+05] Svetlana Kiritchenko, Stan Matwin, A Fazel Famili, et al. Functional annotation of genes using hierarchical text categorization. In *Proc. of the ACL Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, 2005.

[KN19] Zixuan Ke and Vincent Ng. Automated essay scoring: a survey of the state of the art. In *IJCAI*, volume 19, 2019.

[KS97] Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997.

[KT19] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, 2019.

[KW06] Christopher E Keefer and Nathaniel A Woody. Rejecting unclassifiable samples with decision forests. *Chemometrics and intelligent laboratory systems*, 84(1-2), 2006.

[LB+05] Mirella Lapata, Regina Barzilay, et al. Automatic evaluation of text coherence: models and representations. In *IJCAI*, volume 5. Citeseer, 2005.

[LCA+20] Ritchie Lee, Justin Clarke, Adrian Agogino, and Dimitra Giannakopoulou. Improving trust in deep neural networks with nearest neighbors. In *AIAA Scitech 2020 Forum*, 2020.

[LLP+20] Jeremiah Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 33, 2020.

[LLS18] Shiyu Liang, Yixuan Li, and R Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*, 2018.

[LOG+19] Yinhan Liu et al. Roberta: a robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[LPB17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[LPC+15] Stefan Lee, Senthil Purushwalkam, Michael Cogswell, David Crandall, and Dhruv Batra. Why m heads are better than one: training a diverse ensemble of deep networks. *arXiv preprint arXiv:1511.06314*, 2015.

[LSS20] Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2), 2020.

[LXC+18] Maoxi Li, Qingyu Xiang, Zhiming Chen, and Mingwen Wang. A unified neural network for quality estimation of machine translation. *IEICE TRANSACTIONS on Information and Systems*, 101(9), 2018.

[LZM22] Ruiqi Li, Xiang Zhao, and Marie-Francine Moens. A brief overview of universal sentence representation methods: a linguistic view. *ACM Computing Surveys (CSUR)*, 55(3), 2022.

[MA20] Anandhavalli Muniasamy and Areej Alasiry. Deep learning: the impact on future elearning. *International Journal of Emerging Technologies in Learning (Online)*, 15(1), 2020.

[MAN17] AMIT MANDELBAUM. *Distance-based Confidence Score for Neural Network Classifiers*. PhD thesis, Hebrew University of Jerusalem, 2017.

[MG18] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. *Advances in neural information processing systems*, 31, 2018.

[MKS+20] Jooyoung Moon, Jihyo Kim, Younghak Shin, and Sangheum Hwang. Confidence-aware learning for deep neural networks. In *International Conference on Machine Learning*. PMLR, 2020.

[MKvA+23] Jishnu Mukhoti, Andreas Kirsch, Joost van Amersfoort, Philip HS Torr, and Yarin Gal. Deep deterministic uncertainty: a new simple baseline. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[MLG23] Potsawee Manakul, Adian Liusie, and Mark JF Gales. Selfcheckgpt: zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.

[MLZ+17] Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. Interactive attention networks for aspect-level sentiment classification. In *International Joint Conference on Artificial Intelligence*, 2017.

[MNP21] François Monard, Richard Nickl, and Gabriel P Paternain. Statistical guarantees for bayesian uncertainty quantification in nonlinear inverse problems with gaussian process priors. *The Annals of Statistics*, 49(6), 2021.

[MPJ+04] Eleni Miltsakaki, Rashmi Prasad, Aravind K Joshi, and Bonnie L Webber. The penn discourse treebank. In *LREC*. Citeseer, 2004.

[MPR21] Matej Martinc, Senja Pollak, and Marko Robnik-Sikonja. Supervised and unsupervised neural approaches to text readability. *Computational Linguistics*, 47(1), 2021.

[MS99] Christopher D Manning and Hinrich Schutze. Foundations of statistical natural language processing. *Computational Linguistics*, 26(2), 1999.

[MTH+19] Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. Hierarchical text classification with reinforced label assignment. In *Conference on Empirical Methods in Natural Language Processing*, 2019.

[MVH+21] Warren Morningstar, Sharad Vikram, Cusuh Ham, Andrew Gallagher, and Joshua Dillon. Automatic differentiation variational inference with mixtures. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021.

[Nea12] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[NMH+24] Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. Using an llm to help with code understanding. In *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)*. IEEE Computer Society, 2024.

[NMK+17] Jinseok Nam, Eneldo Loza Mencía, Hyunwoo J Kim, and Johannes Fürnkranz. Maximizing subset accuracy with recurrent neural networks in multi-label classification. In *Advances in Neural Information Processing Systems*, 2017.

[NO18] Farah Nadeem and Mari Ostendorf. Estimating linguistic complexity for science texts. In *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications*, 2018.

[NYC15] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: high confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[NZV18] Lukas Neumann, Andrew Zisserman, and Andrea Vedaldi. Relaxed softmax: efficient confidence auto-calibration for safe pedestrian detection, 2018.

[OFR+19] Yaniv Ovadia et al. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in Neural Information Processing Systems*, 32, 2019.

[PBG17] Fabian Pedregosa, Francis Bach, and Alexandre Gramfort. On the consistency of ordinal regression methods. *Journal of Machine Learning Research*, 18, 2017.

[PC19] Dae Hoon Park and Yi Chang. Adversarial sampling and training for semi-supervised information retrieval. In *The World Wide Web Conference*, 2019.

[Ped15] Fabian Pedregosa-Izquierdo. *Feature extraction and supervised learning on fMRI: from practice to theory*. PhD thesis, Université Pierre et Marie Curie-Paris VI, 2015.

[PGK+11] Gaurang Panchal, Amit Ganatra, YP Kosta, and Devyani Panchal. Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers. *International Journal of Computer Theory and Engineering*, 3(2), 2011.

[PLB20] Tim Pearce, Felix Leibfried, and Alexandra Brintrup. Uncertainty in neural networks: approximately bayesian ensembling. In *International conference on artificial intelligence and statistics*. PMLR, 2020.

[PLH+23] Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

[PM18] Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: towards confident, interpretable and robust deep learning. *arXiv e-prints*, 2018.

[PN08] Emily Pitler and Ani Nenkova. Revisiting readability: a unified framework for predicting text quality. In *Conference on Empirical Methods in Natural Language Processing*, 2008.

[PPC+23] Yikang Pan, Liangming Pan, Wenhu Chen, Preslav Nakov, Min-Yen Kan, and William Yang Wang. On the risk of misinformation pollution with large language models. *arXiv preprint arXiv:2305.13661*, 2023.

[PTC+17] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[PYL+22] Francesco Pinto, Harry Yang, Ser Nam Lim, Philip Torr, and Puneet Dokania. Using mixup as a regularizer can surprisingly improve accuracy & out-of-distribution robustness. *Advances in Neural Information Processing Systems*, 35, 2022.

[PZC+08] Rong Pan et al. One-class collaborative filtering. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 2008.

[RA20] Gianna Reggio and Egidio Astesiano. Big-data/analytics projects failure: a literature review. In *2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2020.

[Ras03] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*. Springer, 2003.

[RBL+22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022.

[RF14] Steffen Rendle and Christoph Freudenthaler. Improving pairwise learning for item recommendation from implicit feedback. In *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014.

[RFG+09] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 2009.

[RKX+23] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*. PMLR, 2023.

[RWA+19] Alec Radford et al. Better language models and their implications. *OpenAI Blog https://openai. com/blog/better-language-models*, 1, 2019.

[SBB+16] Kashif Shah, Fethi Bougares, Loic Barrault, and Lucia Specia. Shef-lium-nn: sentence level quality estimation with neural network features. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, 2016.

[SBE+21] Harald Steck, Linas Baltrunas, Ehtsham Elahi, Dawen Liang, Yves Raimond, and Justin Basilico. Deep learning for recommender systems: a netflix case study. *AI Magazine*, 42(3), 2021.

[SF09] Carlos N Silla Jr and Alex A Freitas. A global-model naive bayes approach to the hierarchical prediction of protein functions. In *2009 9th IEEE International Conference on Data Mining*. IEEE, 2009.

[SF11] Carlos N Silla and Alex A Freitas. A survey of hierarchical classification across different application domains. *Data Mining and Knowledge Discovery*, 22(1-2), 2011.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 2014.

[SKK+02] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *5th International Conference on Computer and Information Science*, volume 1 of number 012002. Citeseer, 2002.

[SKK18] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *Advances in Neural Information Processing Systems*, 31, 2018.

[SL01] Aixin Sun and Ee-Peng Lim. Hierarchical text classification and evaluation. In *Proceedings 2001 IEEE International Conference on Data Mining*. IEEE, 2001.

[SL09] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information processing & management*, 45(4), 2009.

[SLL+23] Xiaofei Sun et al. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

[SS04] Awdhesh Kumar Singh and Rajendra Sahu. Decision support system for hs classification of commodities. In *Proceedings of the 2004 IFIP International Conference on Decision Support Systems (DSS 2004)*, 2004.

[STC+09] Lucia Specia, Marco Turchi, Nicola Cancedda, Nello Cristianini, and Marc Dymetman. Estimating the sentence-level quality of machine translation systems. In *EAMT*, volume 9, 2009.

[STQ+19] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mass: masked sequence to sequence pre-training for language generation. In *International Conference on Machine Learning*. PMLR, 2019.

[SUV18] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018.

[SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 27, 2014.

[TAS18] Mattias Teye, Hossein Azizpour, and Kevin Smith. Bayesian uncertainty estimation for batch normalized deep networks. In *International conference on machine learning*. PMLR, 2018.

[TCB+19] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: improved calibration and predictive uncertainty for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[TKY+24] Ryan Theisen, Hyunsuk Kim, Yaoqing Yang, Liam Hodgkinson, and Michael W Mahoney. When are ensembles really effective? *Advances in Neural Information Processing Systems*, 36, 2024.

[TL19] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[TLI+23] Hugo Touvron et al. Llama: open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[TN16] Kaveh Taghipour and Hwee Tou Ng. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016.

[TO19] Sho Takase and Naoaki Okazaki. Positional encoding to control output sequence length. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019.

[TPT+18] Yi Tay, Minh C Phan, Luu Anh Tuan, and Siu Cheung Hui. Skipflow: incorporating neural coherence features for end-to-end automatic text scoring. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[TVN+22] Kalliopi Tsolaki, Thanasis Vafeiadis, Alexandros Nizamis, Dimosthenis Ioannidis, and Dimitrios Tzovaras. Utilizing machine learning on freight transportation and logistics applications: a review. *ICT Express*, 2022.

[US19] Venkatesh Umaashankar and Girish Shanmugam S. Multi-label multi-class hierarchical classification using convolutional seq2seq. In *KONVENS*, 2019.

[vASJ+21] Joost van Amersfoort, Lewis Smith, Andrew Jesson, Oscar Key, and Yarin Gal. On feature collapse and deep kernel learning for single forward pass uncertainty. *arXiv preprint arXiv:2102.11409*, 2021.

[VFB95] Germano C Vasconcelos, Michael C Fairhurst, and David L Bisset. Investigating feedforward neural networks with respect to the rejection of spurious patterns. *Pattern Recognition Letters*, 16(2), 1995.

[VHdC+07] Diego P Vivencio, Estevam R Hruschka, Maria do Carmo Nicoletti, Edimilson B dos Santos, and Sebastian DCO Galvao. Feature-weighted k-nearest neighbor classifier. In *2007 IEEE Symposium on Foundations of Computational Intelligence*. IEEE, 2007.

[VIY20] Nurali Virani, Naresh Iyer, and Zhaoyuan Yang. Justification-based reliability in machine learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34 of number 04, 2020.

[VKT+23] Artem Vazhentsev et al. Hybrid uncertainty quantification for selective text classification in ambiguous tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023.

[VSP+17] Ashish Vaswani et al. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[VST+20] Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*. PMLR, 2020.

[WCB18] Jonatas Wehrmann, Ricardo Cerri, and Rodrigo Barros. Hierarchical multi-label classification networks. In *International Conference on Machine Learning*. PMLR, 2018.

[WH00] Rüdiger Wirth and Jochen Hipp. Crisp-dm: towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, volume 1. Manchester, 2000.

[WHZ+16] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based lstm for aspect-level sentiment classification. In *Empirical Methods in Natural Language Processing*, 2016.

[WLS20] Chenguang Wang, Xiao Liu, and Dawn Song. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*, 2020.

[WMR+21] Laura Weidinger et al. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359*, 2021.

[WS17] Zhenzhou Wu and Sean Saito. Hinet: hierarchical classification with neural network. *arXiv preprint arXiv:1705.11105*, 2017.

[WST+20] Florian Wenzel, Jasper Snoek, Dustin Tran, and Rodolphe Jenatton. Hyperparameter ensembles for robustness and uncertainty quantification. *Advances in Neural Information Processing Systems*, 33, 2020.

[WXC+22] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. In *International Conference on Machine Learning*. PMLR, 2022.

[WYY+06] Xie Wei, Li Yinsheng, Xu Yingxiao, and Ma Zhanxin. Implementing knowledge base for hs matchmaking. In *2006 IEEE International Conference on e-Business Engineering*. IEEE, 2006.

[WZ89] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2), 1989.

[XAZ+19] Chen Xing, Sercan Arik, Zizhao Zhang, and Tomas Pfister. Distance-based learning from errors for confidence calibration. In *International Conference on Learning Representations*, 2019.

[XB22] Guoxuan Xia and Christos-Savvas Bouganis. Augmenting softmax information for selective classification with out-of-distribution data. In *Proceedings of the Asian Conference on Computer Vision*, 2022.

[XB23] Guoxuan Xia and Christos-Savvas Bouganis. Window-based early-exit cascades for uncertainty estimation: when deep ensembles are more efficient than single models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[YDY+19] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, 2019.

[YKG+24] Yasin Abbasi Yadkori, Ilja Kuzborskij, András György, and Csaba Szepesvári. To believe or not to believe your llm. *arXiv preprint arXiv:2406.02543*, 2024.

[YSL+18] Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. Sgm: sequence generation model for multi-label classification. In *International Conference on Computational Linguistics*, 2018.

[ZA20] Weimin Zhou and Mark A Anastasio. Markov-chain monte carlo approximation of the ideal observer using generative adversarial networks. In *Medical Imaging 2020: Image Perception, Observer Performance, and Technology Assessment*, volume 11316. SPIE, 2020.

[ZCD+18] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[ZCW+13] Weinan Zhang, Tianqi Chen, Jun Wang, and Yong Yu. Optimizing top-n collaborative filtering via dynamic negative item sampling. In *Proceedings of the 36th international ACM SIGIR Conference on Research and Development in Information Retrieval*, 2013.

[ZCZ+22] Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-Lin Liu. Rethinking confidence calibration for failure prediction. In *European Conference on Computer Vision*. Springer, 2022.

[ZCZ+23] Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-Lin Liu. Openmix: exploring outlier samples for misclassification detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[ZDM20] Changxuan Zhao, Mahmoud Dinar, and Shreyes N Melkote. Automated classification of manufacturing process capability utilizing part shape, material, and quality attributes. *Journal of computing and information science in engineering*, 20(2), 2020.

[ZFW+17] Xiao Zhang, Zhiyuan Fang, Yandong Wen, Zhifeng Li, and Yu Qiao. Range loss for deep face recognition with long-tailed training data. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[ZLL+18] Min-Ling Zhang, Yu-Kun Li, Xu-Ying Liu, and Xin Geng. Binary relevance for multi-label learning: an overview. *Frontiers of Computer Science*, 12(2), 2018.

[ZLZ+19] Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. In *International Conference on Learning Representations*, 2019.

[ZXL+23] Xu-Yao Zhang, Guo-Sen Xie, Xiuli Li, Tao Mei, and Cheng-Lin Liu. A survey on learning to reject. *Proceedings of the IEEE*, 111(2), 2023.

[ZZE+21] Sheheryar Zaidi, Arber Zela, Thomas Elsken, Chris C Holmes, Frank Hutter, and Yee Teh. Neural ensemble search for uncertainty estimation and dataset shift. *Advances in Neural Information Processing Systems*, 34, 2021.

[ZZH+22] Qiannan Zhu, Haobo Zhang, Qing He, and Zhicheng Dou. A gain-tuning dynamic negative sampler for recommendation. In *Proceedings of the ACM Web Conference 2022*, 2022.

Appendix

List of Figures

1.1 Selective classification example: Using the distance to decision boundary to reject samples. The confidence score $C(x)$ here is the distance to the decision boundary. Without any rejection, the model can achieve 60% precision with 100% recall. However, if we apply the rejection as shown in the figure, the model achieves 87.5% precision with 53.3% recall. 2
1.2 Precision-recall plot: The tradeoff between between precision and recall performance of the model. A better rejection approach should have higher recall under the same precision. Model A performs better before the intersection point while Model B performs better after the intersection point. 4
1.3 A toy example of aleatoric and epistemic uncertainty. 6
1.4 Optimization difference between cost-known vs cost-unknown. . 6
1.5 CRISP-DM methodology 12

2.1 Different types of confidence score: softmax and its variations, nearest neighbor, density evaluation, and ranking. The baseline and its regularization use the maximal value of the softmax to represent the confidence score. The nearest neighbor based approaches use the distance or the amount of nearest neighbor to represent its confidence in hidden layer(s) or last layer. The density-based approaches calculate the density score in its training data to represent the confidence score, and the ranking-based approaches use different targets that are derived from training data/process to represent the confidence score. 17

3.1 HS Code illustration 30
3.2 Hierarchical Model 32
3.3 HS code distribution 33
3.4 NMT Architecture 34
3.5 LSTM model structure 35
3.6 Long time short memory 36

4.1 Pipeline for importer prediction. 41
4.2 Exact match mapping extraction 45
4.3 Beam search on rules. 45
4.4 Similarity feature. 46
4.5 Dynamic sampling in the importer prediction task. 47
4.6 True class probability Training process. Model 1 is a probabilistic binary classifier, while model 2 will calibrate the predicted confidence score. After the model is trained, the sample will feed into both model 1 and model 2 sequentially, the final output is the predicted probability. 48
4.7 Precision-recall plot on test data. 51

5.1 Visualisation of the differences between True Class Probability (TCP) [CTB+19], Failure Detection (FD) [HDV18] and our proposed Confidence Range (CR) method. 55
5.2 Overview of our Confidence Range architecture. After training a classifier, we train an auxiliary confidence network and use the classifier's output to define our target confidence range. . . 62
5.3 Rejection option comparison: X is the precision, Y is the improvement in terms of coverage, expressed as coverage(Evaluation approach) - coverage(MSP). A Negative value means that the approach is worse than the baseline. 65
5.4 Comparison between LSTM-based NMT and Transformer for both MSP and CR option. In this plot, the diff is the Transformer's coverage minus the LSTM-based NMT's coverage. . . 67

6.1 Label distribution 75
6.2 Model architecture 75
6.3 Illustration of domain knowledge. The description will be converted to a feature vector with binary value $[1, 0, 1, 0, 0, 0]$. . . . 76
6.4 Score distribution 78

7.1 Data structure 80
7.2 Pre-train target. 82
7.3 The precision-recall plot. 84

7.4 A violin plot is used to visualize the distribution of epistemic uncertainty in ”Correct Predictions” and ”Incorrect Predictions.” The width of the violin at different levels of the vertical axis represents the density of the data, with wider sections indicating a higher density of values at that range. In theory, misclassified samples (right side) tend to have higher epistemic uncertainty (e.g., outliers, novelty, etc.) compared to correctly classified samples (left side). However, the distribution in the higher uncertainty region shows a similar density for both categories. This suggests that the model struggles to make decisions in those areas. Combined with sample checks, this indicates the randomness in customer behavior when deciding whether to call customer service or not. 86

8.1 Example of ChatGPT-4 in classifying description quality . . . 93
8.2 Pathway to impact 95

List of Tables

2.1 Summary of Softmax-related approaches 18
2.2 Summary of nearest neighbor approaches 20
2.3 Summary of single uncertainty-related approaches 22
2.4 Summary of Aleatoric and Epistemic uncertainty related approaches 24
2.5 Summary of ranking based approaches 25
2.6 Summary of ensemble related approaches 27

3.1 Model comparison on accuracy and recall. The first columns is the desired accuracy, it is used for calculating the threshold for the confidence score on the validation dataset. The accuracy for each model is the accuracy where its probability is larger than the threshold. 37

4.1 Exact rule matching results 49
4.2 Recall under different precision requirements 49
4.3 Paired T-test on FWNN and our proposed models 50

5.1 Data set statistics 64
5.2 Selective classification coverage results on the three selected data sets, using precision to specify the rejection threshold. . . 65
5.3 The p-value of Student's t-test on MSP and CR approach. . . . 66
5.4 Results of the ablation study where we add or remove different components of our model as well as its effects on the two different model architectures. 68
5.5 The statistical significance of the performance differences in our ablation study. 68
5.6 Performance comparison on image classification data (CIFAR-10 and CIFAR-100). 69

6.1 The statistics of the dataset 75
6.2 Results 78

7.1 The proportions of the last event before the customer called with the non-call shipments for the same event. We show only the top 4 events related to customer calls. 84
7.2 Recall under the different precision threshold. The number in the bracket is the P-value from paired student T-test compared with Transformer. A precision threshold can to set in order to decide which customers should be updated. 85