Unsupervised Brand Name Extraction Using Domain Adaptation

by

Afsaneh Towhidi

A thesis submitted in partial fulfillment
of the requirements for the degree of

Masters of Applied Science

in

Electrical and Computer Engineering

University of Ontario Institute of Technology

Oshawa, Ontario, Canada

August 2019

THESIS EXAMINATION INFORMATION

Submitted by: **Afsaneh Towhidi**

Masters of Applied Science in **Electrical and Computer Engineering**

Thesis title: Unsupervised Brand Name Extraction Using Domain Adaptation

An oral defense of this thesis took place on August 12, 2019 in front of the following examining committee:

Examining Committee:

Chair of Examining Committee	Dr. Ying Wang
Research Supervisor	Dr. Masoud Makrehchi
Research Co-Supervisors	Dr. Shahryar Rahnamayan
Examining Committee Member	Dr. Akramul Azim
Thesis Examiner	Dr. Masoud Makrehchi

The above committee determined that the thesis is acceptable in form and content and that a satisfactory knowledge of the field covered by the thesis was demonstrated by the candidate during an oral examination. A signed copy of the Certificate of Approval is available from the School of Graduate and Postdoctoral Studies.

Abstract

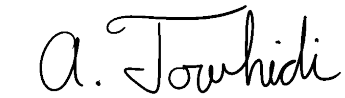
Business intelligence and analytics is an area of research that analyzes the existing business data to extract the insights needed for a successful business planning. Textual data derived from tweets, forum posts, and blogs are from different business domains, and contain useful information for the organizations. This thesis proposes a method for extracting brand and product names from text; brand names as a subset of named entities can give a great deal of information about the whole document.

In this thesis, a context window is defined to capture the context of a word in a sentence. In addition, a word embedding model is locally trained to have a domain specific model and finally, a domain adaptation technique is employed to transfer the knowledge from one domain with labeled data to a new domain. The results indicate a significant improvement in recall measure for extracting brand names from a new domain.

Keywords: Natural Language Processing, Named Entity Recognition, Word Embedding, Domain Adaptation

Author's Declaration

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I authorize the University of Ontario Institute of Technology to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize University of Ontario Institute of Technology to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

Signed: 

Afsaneh Towhidi

Statement of Contribution

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication. I have used standard referencing practices to acknowledge ideas, research techniques, or other materials that belong to others. Furthermore, I hereby certify that I am the sole source of the creative works and/or inventive knowledge described in this thesis.

To Ahmad, Mehrandokht, Nima

"To have been loved so deeply will give us some protection forever."

- Albus Dumbledore, J. K. Rowling

Acknowledgements

I would like to express my deepest appreciation to my advisors Dr. Masoud Makrehchi and Dr. Shahryar Rahnamayan for guiding me throughout my research and providing me with valuable and constructive suggestions during the planning and development of this research work. It was a privilege and an honor to work with you.

The completion of this study would not have been possible without the support and nurturing of Dr. Annie Lee. You played a decisive role in shaping my view on research and the importance of being precise and diligent. Thanks for providing me with encouragement and patience throughout the duration of my research.

I am also grateful to my friends and colleagues in Social Computing and Collective Intelligence Lab (SciLab) at Ontario Tech University: Tara Ahmadalinezhad, Mehran Kamkarhaghighi, and Fateme Azimlou. Thanks should also go to my friends in Nature Inspired Computational Intelligence Lab (NICI) at Ontario Tech University with their insightful comments and encouragement during our weekly meetings.

Especial thanks are due to David, Doris, and Steven whose help cannot be overstated. You are my home away from home.

Finally, I am especially grateful to my parents, Ahmad and Mehrandokht, who supported me emotionally and financially. I always knew that you believed in me and wanted the best for me. Thank you for teaching me that my job in life was to learn, to be happy, to enjoy my work and to know and understand myself. A very especial

thanks goes to my brother, Nima, who offered invaluable support and humor over the years. Without your help, this would not have been possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The amount of text data generated through the internet has grown exponentially over the past three decades. Organizations and companies can use the information extracted from this data to address challenges and opportunities. Business Intelligence and Analytics (BI&A) is a challenging area of research in both industry and academia. From a business's point of view, organizations are more inclined to gain access to the information they need by spending the least amount of money. Therefore, an information extraction system that can work for different products and business domains is the ideal approach. However, this is a challenging task for researchers because the information and patterns that one can extract from the text in one business domain is often different from other information. This is even more challenging for machine learning researchers, since the most important part of modeling is training datasets with a significant amount of labeled data; labeling data is expensive and time-consuming. In addition, most of the time, a human annotator and/or domain expert are needed.

Information Extraction (IE) is one of the tasks in Natural Language Processing (NLP), and NER, as one of the sub tasks of IE, plays an important role in it [1].

Therefore, improving the performance of NER systems might improve the performance of overall IE systems. In the business world, the brand and product names used in sentences can give a great deal of information about the textual data. Moreover, the extracted brand names can be used in market analysis, business intelligence, IE, the creation of brand catalogues, and sentiment analysis of products and brands. For instance, it may help a company extract information from social media, and analyze the needs and opinions a target society to make proper decisions and increase profits. Brand and product names can be represented with a proper noun, therefore, brand name extraction is a subset of NER. This field of study is important in the competitive business world, and the first serious discussion and analysis of NER regarded extracting company names from financial news stories [2].

Textual data can come from different domains. People can post tweets, write blogs and reviews, and add comments about automotive, apparel, technology, etc. Using traditional machine learning approaches, such as supervised learning, is not always feasible due to the high cost of generating training data and also the evolving nature of named entities, which causes some entities to appear in the evaluation that have not been seen in the training data.

To overcome this challenge, two methods are used in this thesis: Domain adaptation, and word embedding. In domain adaptation, knowledge from one domain can be transferred to other domains. Hence, with the labeled data of one domain, it is possible to label other domains' data. The context of words can carry significant knowledge about a domain, and for this purpose, word embedding can be employed. Word embedding is a form of word representation. Word embedding has received considerable attention in recent decades, and many deep learning-based word embedding approaches such as GloVe, Word2Vec, fastText, and BERT are used in different studies.

This thesis consists of five chapters. Following the introduction in Chapter 1, a review of NER is provided in Chapter 2 with the problem statement at the end of the chapter. Chapter 3 describes the proposed approach for extracting brand names. Chapter 4 the experiments and results are discussed. Chapter 5 gives a brief summary and critique of the findings and suggests future areas of research.

Chapter 2

Named Entity Recognition

2.1 Introduction

A named entity is a unit of information such as a person, organization or location [3]. NER is a key component in many natural language processing applications such as automatic forwarding, question-answering systems, document searching, and text summarization.

One of the first discussions and analysis of named entity extraction emerged during the 1990s with Lisa F. Rau's paper, which focused on extracting company names from textual data [2]. By relying on the concepts of rule-based and heuristic methods, Rau was able to extract the company names from financial news with 95% accuracy. In recent years, there has been an increasing amount of literature on NER and the importance of this field in other NLP tasks.

In this chapter, a background of NER is provided. The applications of NER are discussed in section 2.2, followed by section 2.3 which discusses the challenges that an NER system may face. The traditional machine learning approaches which can be used in an NER system are reviewed in section 2.5, where the evaluation of these

methods is discussed in section 2.6.

2.2 Applications of Named Entity Recognition

NER is one of the most important subtasks of information extraction [1]. It is also a tool for pre-processing texts in a myriad of important NLP tasks, and in most cases, it increases the performance significantly. In this section, some of the applications of NER are discussed.

2.2.1 Question answering systems

Automatic Question Answering (QA) systems are developed to generate answers from structured and unstructured data for human questions. QA systems should handle different types of questions: questions with Yes/No answers, factoid questions, and multiple choice questions.

A major type of question in QA systems are factoid type questions, the answers to which are concise facts. Factoid questions are 'wh-words' (what, when, whom, why, where, how long, how far, etc.) and can generally be answered in short texts. Hence, extracting Named Entitys (NEs) can be advantageous in answering these types of questions. The answer to "who" is a PERSON, "when" is TIME or DATE, "how far" is LENGTH, "how rich" is MONEY, and so on. By applying a named entity tagger on TREC-8 QA track competition, R. Srihari et al. achieved 66.0% accuracy [4].

However, there is always ambiguity in labeling entities. Some entities can have different labels in different situations. For example, "Dell" can be both a company named entity and a person named entity as its founder, "Michael Dell". Traditional NER usually supports one label per named entity. D. Molla et al. demonstrated that assigning multiple labels to a named entity and letting the QA system decide the

correct label with further investigations would eventually increase the recall [5].

2.2.2 Machine Translation

Machine Translation (MT) refers to an automated translation of text or speech from one language into another language [6]. Named entities are an important part of MT, and the incorrect labeling can cause serious problems in the system. It is crucial to differentiate common nouns from named entities because failing to do so may decrease the performance of the translation system. B. Babych et al. concluded that information extraction techniques (including NER) can increase the quality of output in MT systems [7]. Similarly, Y. Chen et al. showed that proper named entity identification and named entity alignment between Chinese and English can increase the result significantly: 21.3% relative improvement in F-score [8].

2.2.3 Opinion mining

In the world of social media and blogs, people share their opinions on different organizations, products, and events; and have a chance to read others' opinions. Analyzing the opinions of customers can make an immense difference in the profit of organizations or the career success of individuals. Opinion mining, or sentiment analysis, is the process of extracting peoples' opinions about products, services, events, organizations, and in general about entities [9].

Most studies regarding NER focus on lengthy textual data. However, with the advent of Web 2.0, the studies on micro-text such as tweets and comments attain more attention [10]. With this intention, Jung et al. improved the online NER task by proposing a new approach for clustering the microtexts based on different contextual associations, which improved the precision of extracting information significantly [11].

Another research on employing NER in opinion mining systems is the work from Spina et al., where they defined filter keywords, such as the words in a tweet, that can accurately argue whether the tweet indicates a company name [12]. Spina proposed a method for company name disambiguation by identifying filter keywords without using any annotated data related to the target company. Classifying the tweets using these keywords, a machine learning algorithm can then be applied on all tweets. In comparison to other supervised approaches, this unsupervised classification achieved a promising result.

2.2.4 Semantic search

Semantic search focuses on improving the search experience by trying to learn the user's intention which results in more personalized, reliable and relevant answers. Furthermore, the semantic search can then be used in Decision Support Systems (DSS) which have applications in Business Intelligence, Information Retrieval, emergency management, etc. [13].

I.Habernal et al. [14] demonstrated the necessity of extracting named entities from texts in semantic search. They added relative location and relative price words, such as near, cheap, and internet, to the traditional named entity categories.

2.3 Challenges

NER like any other NLP tasks requires prior knowledge to achieve a complex prediction. NER has many challenges such as nested entities, data annotation, capitalization issues, spelling diversity, etc. Some important challenges are discussed in this section.

Nested entities Named entities may contain other named entities. Due to technological reasons, nested entities are usually ignored by NER tools. However, recognition of nested entities may improve the overall performance of the system significantly [15].

Segment labeling, also known as chunk tagging, is one of the approaches that can handle nested entities [16, 17].

Ambiguity One named entity can refer to various entities. For instance, Python can be a name of programming language, snake or a movie. Tackling this issue is called Named Entity Disambiguation (NED) which is a main component in Entity Linking (EL) [18]. One of the approaches for NED is to analyze the contextual information where the word appears.

Annotation Annotating the training data for supervised learning methods is remarkably time-consuming and expensive. Moreover, since the documents are in different domains, domain experts are needed for annotating each different document.

One of the main approaches for this issue is annotating the data using semi-supervised learning algorithms to achieve a large annotated dataset using a slight number of annotated data [19, 20].

Availability of Resources Although there are numerous resources in English such as gazetteers, POS taggers, etc., that may help finding named entities, most of the languages lack resources for NER systems and, therefore, the NER classification task is more challenging in these languages [16].

2.4 Features

Feature engineering is one the most preeminent tasks in machine learning based NER whereas in rule based NER, features are not collected due to the large volume of data and the complexity of formats and diverse content. Features describe the characteristics of the words and can be represented by Boolean, numeric, or nominal values [21]. For instance, a Boolean value can be used as a feature and set to true if the word contains digits, and false otherwise. Similarly, the length of the word can be employed as a numeric feature, and the stem of the word as nominal feature.

In this section, the word level features and representation learning are discussed.

2.4.1 Word level features

Word level features describe the characteristics of the word in the character level. Some examples of word level features are provided in Table 2.1. Digit patterns and functions from the table are discussed in the following.

Digit patterns Dates, percentage, and intervals are some of the examples of digit patterns that may give a cue about the named entities. There are certain patterns for showing the dates (two digits for days and months and four digits for years) and digits followed by % symbol can stand for percentage [21]. These patterns can also be employed to categorize named entities.

Functions Applying a function on words can create a feature set. For instance, an n-gram function (discussed in section 2.4.2) can be applied on the document to create an n-gram feature set. Another approach is to apply shape functions to map the words to predefined patterns.

Features	Case	Examples
Case	- w_i is all uppercased/lowercased/mixed case	APPLE, apple, eBay
	- w_i starts with a capital letter	Apple
Punctuation	- w_i has period at the end or in the middle	Dr., C.N.N., Corp.
	- w_i has apostrophe, hyphen, or ampersand in the middle	O'Malley
Digit	- w_i contains digits	T120
	- w_i is cardinal or ordinal	one, first
	- w_i is a Roman number	VI, IX
	- w_i has a digit pattern	1989, 80s, 50Kg
Part-of-speech	- The categorization of w_i as noun, pronoun, verb, adverb, etc.	
Morphology	- Prefixes and suffixes of length n in w_i	un-, -ful, -ness
	- w_i stem	fishing and fished, stem: fish
	- w_i has similar ending with other known words	psychotherapist, journalist, Nagpur, Jaipur
Function	- n-gram of w_i	
	- Shape and short shape of w_i	
	- w_i length	
Character	- w_i contains any special symbol	student's, Θ

Table 2.1: Given a sequence of words in a sentence: ..., $w_i-2, w_i-1, w_i, w_i+1, w_i+2$, the suggested features for the current word w_i has been shown in the table

2.4.2 Representation learning

Using words in a document to create document representation is an intuitive approach in NLP. A set of unique words w can create a dimensional feature space d in which text T can be represented as a point [22]. For instance, $T_w = [w_1, w_2, ...w_d]$ is the representation vector that does not keep the word order in the text; $d = R^k_{size}$ is based on the Heap's law, $k \in < 0.3, 0.7 >$ which is set using the language, and R_{size} is the size of the document. This representation is called bag-of-words. Bag-of-words, first introduced in 1988 by Salton and Buckley [23], is a document representation with a fixed-length vector; each component in the vector represents one word in the document and the length of the vector is the number of unique words. The d value may be very large when it is calculated for a large document. Hence, some pre-processings, such as stop words filtering, stemming, and lemmatization, may help reducing the d value.

Based on a similar idea, n-gram was defined as $T_{nw} = [nw_1, nw_2, ..., nw_d]$ where nw is a set of n unique neighbouring words in the document [22]. Therefore, as it is shown in Fig. 2.1, instead of using single terms in the representation, chunks of n words can be employed to capture the semantic of the text.

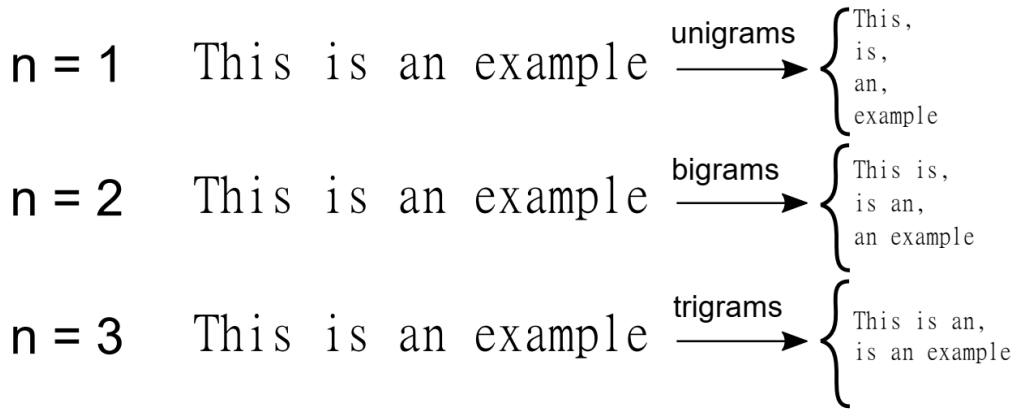


Figure 2.1: N-Gram representation when n = 1,2,3

Generally, high dimensionality or "curse of dimensionality" has always been an immense challenge in learning problems. Bag-of-words and n-gram were two examples of traditional document representations in which the representation suffers from the curse of dimensionality. Most of the current research on document representation pays particular attention to word embeddings in which words or phrases are mapped to vectors with optional and usually low dimensionality. In this section, a review of word embedding is presented.

Word Embedding

Word embedding which was first proposed in 1989 is a language model that maps words or phrases to corresponding vectors [24]. In other word, it is defined as the mapping of $V \rightarrow \mathbb{R}^D : w \longmapsto \overline{w}$ where w is a word, V is the vocabulary, $\overline{w}$ is a real-valued vector, and D is the dimensionality of the embedding space [25].

Beside the significantly reduced dimensionality of feature space in word embedding compared to the traditional document representations, word embedding has two major deficiencies: Out Of Vocabulary (OOV) words and ambiguity.

OOV words are the words which did not appeared in the text that was used in the training of word embedding model and, therefore, there are no mapping vectors for these words in the model. Researchers have investigated a variety of approaches to handle OOV. Some of these approaches are as follows:

- Zero padding: Assign zero vector to the OOV word with the same dimensionality as other vectors in the model.
- Random vector: Assign a vector with random noise to the OOV word and store it for its next occurrence [26].

If a word is not OOV, the other issue that may come up is ambiguity. While vectors

are point estimates, the word in the document may have different senses. Therefore, finding the right sense is crucial before assigning the corresponding vectors. Using the word context is one of the approaches to find the right sense for each word.

There are different models that can be used to create a word embedding; Word2Vec [27], GloVe [28], fastText [29], and BERT [30] are some of the most used models in research and industry. In this study, Word2Vec is employed and hence, a review of this model is presented in the next section.

Word2Vec

Word2Vec is a distributed representation of words learned by a two-layer neural network-based approach. First introduced by Mikolov et al. [27] in 2013, Word2Vec model uses two architectures, Continuous Bag of Words (CBOW) and Skip-gram, to train word vector representation. Iterating over the corpus, CBOW aims to predict the current word using the surrounding words. On the opposite side, skip-gram tries to predict the surrounding words using the current word.

The training complexity for both CBOW and skip-gram is:

$$O = E \times T \times Q, \tag{2.1}$$

where E specifies the number of training epochs, T shows the number of words in the training set, and Q is the computational complexity per each training in the architecture [27].

Continuous Bag of Words CBOW model focuses on predicting the current word using the related context which in this case is n words from the history and n words from the future. As shown in Fig.2.2 [27], CBOW predicts the current word using $w(t-2)$, $w(t-1)$ from history and $w(t+1)$ and $w(t+2)$ from the future. After

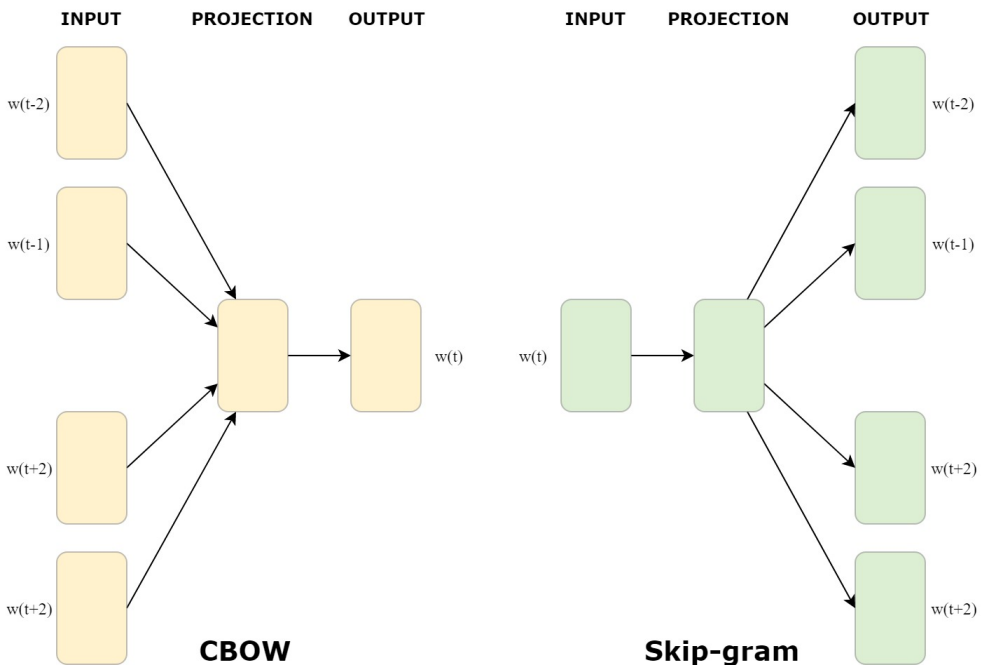


Figure 2.2: Different architectures of CBOW and Skip-gram

counting the frequency of each word in the training corpus, the CBOW model applies Huffman encoding to be able to show each word with "0" and "1".

This model is a bag-of-words model because first, the order of words in the history is not of paramount importance and second, the context has a continuous distributed representation.

The Q in equation 2.1 for CBOW model is

$$Q = N \times D + D \times log_2(V),$$

where N is the number of inputs, D is word representations, and V is dimensionality [27].

2.5 Machine Learning Methods used in NER

Machine learning is the science that helps computer systems automatically learn complex patterns from the data in order to operate a particular task on the data without any predefined rules. In this section, three machine learning methods are discussed: supervised learning, semi-supervised learning, and unsupervised learning. These methods can be applied to the data when the source dataset and target dataset are from the same domains. At the end, a brief introduction to domain adaptation is presented. Domain adaptation can work when the source and target dataset are from different domains.

2.5.1 Supervised Learning

Supervised learning is a form of classification in which feature set and labeled data are needed to train the model. In NER, the named entities in training data are labeled manually by experienced human annotators and the annotation is often expensive and time-consuming.

One of the most important tasks in supervised learning is selecting the relevant features to help the model to find the patterns between similar data. An explanation on features that can be used in NER tasks is given in section 2.4.

There are different learning algorithms that can be used for NER tasks. Hidden Markov Model (HMM) [31], Support Vector Machine (SVM), and Conditional Random Field (CRF) are some of these algorithms.

2.5.2 Unsupervised Learning

Unlike supervised learning, there is no labeled data in unsupervised learning approaches. The idea is using data distribution and hidden patterns to gain knowledge

about the data.

The most common unsupervised learning task is clustering [32,33]. In NER tasks, context similarity and distributional statistics can be employed to label the named entities.

2.5.3 Semi-supervised Learning

It was noted in section 2.5.1 that creating a reliable annotated data is expensive, time-consuming, and sometimes unattainable. To overcome this deficiency, semi-supervised learning used a large number of unlabeled data and the limited provided labeled data. This approach employs the knowledge from the labeled data to label unlabeled data. As a result, more labeled data are generated that can then be used to train the model.

Bootstrapping [34], self-training and co-training [35,36], and graph-based methods [37] are some of the most popular methods of semi-supervised learning used in NER systems.

2.5.4 Domain Adaptation

In the previous mentioned learning methods, the assumption is that the training data and test data are from the same domain. However, having access to labeled training data is not always feasible and, therefore, the assumption does not hold. Transfer learning and domain adaptation are the methods that can be employed in these scenarios and are explained in section 3.2 in detail.

2.6 Evaluation Methods

Like any other machine learning tasks, the evaluation of an NER system plays an important role in their development. The correct labels are usually annotated by a linguist or someone who is the domain expert. The evaluation is the comparison between the correct labels and the system's predicted labels to check the performance of the system.

In this section, two most used evaluation in NER systems are discussed.

2.6.1 Recall, precision, and F-score

Precision, Recall, and F-score are calculated using the parameters that are shown in Figure 2.3. True Positive (TP) are the cases in which the labels are correctly predicted as positive, while False Positive (FP) are the cases in which the labels are predicted positive while the actual label were negative. Furthermore, False Negative (FN) are when the actual labels are positive but the model predicted them as negative, and True Negative (TN) are the cases in which the model are correctly predicted the labels as negative. Precision and recall are calculated by the formulas given below.

$$Precision = \frac{TP}{TP + FP} = \frac{\text{Number of correct positive predictions}}{\text{Total number of positive predictions}} \tag{2.2}$$

$$Recall = \frac{TP}{TP + FN} = \frac{\text{Number of correct positive predictions}}{\text{Total number of instances that should have been classified as positive}} \tag{2.3}$$

F-score takes both FP and FN into account since it is the harmonic average of

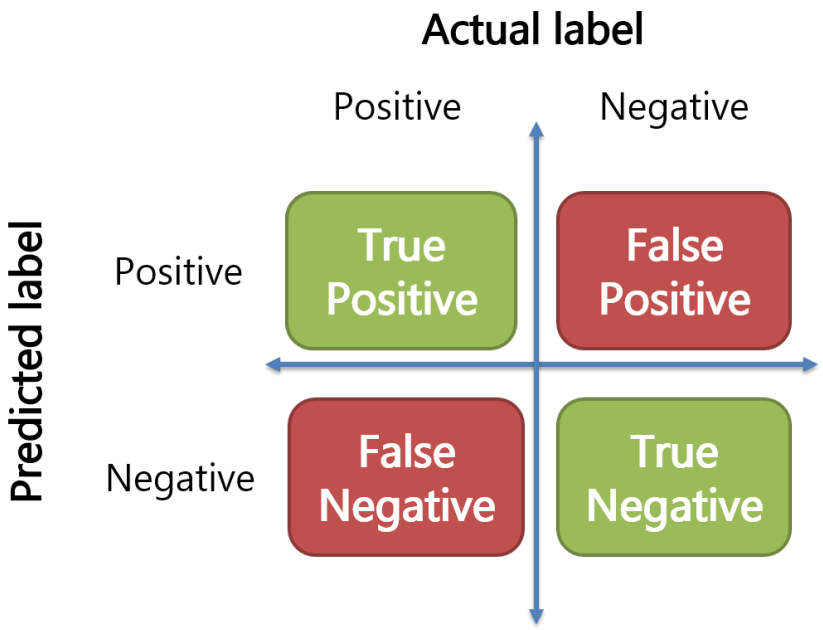


Figure 2.3: Confusion matrix demonstrates the performance of a classification model

precision and recall. With this definition, one can compute F-score:

$$\text{F-score} = 2 * \frac{Precision.Recall}{Precision + Recall} \tag{2.4}$$

2.6.2 Cross-validation

Cross-validation is an easy implemented statistical method to evaluate the performance of a machine learning model. This evaluation method is commonly used for supervised learning approaches.

This technique, also called k-fold cross-validation, splits the randomly shuffled dataset into k chunks and train a model based on *k-1* chunks and then evaluate it on the remained chunk. This training is repeated for k iterations and the error estimation is the average of all k trial scores. Hence, every instance has a chance to be in the training data *k-1* times and once in the test data. Therefore, cross-validation

results in a lower bias evaluation compared to the simple train/test split as all the data is being used in the training phase. In addition, the variance may be lowered accordingly as all the data is being employed in the test set.

2.7 Brand Names Extraction

A brand name can be represented with a proper noun. Therefore, brand names are named entities and brand name extraction is a subset of Named Entity Extraction. Extracting brand names plays an important role in business intelligence, information extraction, and creation of brand catalogues. Furthermore, it may help the companies to extract information from social media and analyze the need and opinion of the target society to make proper decisions and increase the profit. Accordingly, The first systematic study of NER was done by Rau in 1991 for extracting company names from financial news stories [2].

There are numerous studies on extracting brand and company names extraction, and brand analysis solutions. Spranger et al. [38] proposed COBRA, brand and reputation analysis solution, that monitors massive amount of Consumer Generated Media (CGM), such as message boards and forums, content and alerts brand and reputation issues. In the same vein, Mostafa [39] analyzed 3516 tweets for sentiment polarity towards well-known brands. In his paper, he points out that customer relationship management, text filtering, and public opinion tracking are some of the fields that can benefit from sentiment mining. Another study on brands is brand associations which is anything that comes to customer's mind about the brand. Malouf's [40] investigated the quantitative techniques for extracting brand associations of eight leading medications for seizure disorder. He noted that the basic sentiment analysis is not useful for medical brand association extraction since most writings related to

seizure's medications are negative.

All things considered, brand and product names have different characteristics. A brand name may:

- Starts with an uppercase letter: Apple, Honda
- Contains punctuations: Coca-Cola, Yahoo!, Toys"R"Us
- Contains digit(s): 3M, 7 Up, 7-Eleven
- Contains all uppercased letters: BMW, LEGO, IBM
- Be in the position of subject or object in the sentence.
- Be in the verb position: You can google this rule.

In NLP, a text preprocessing step is always needed before data exploration and modeling. The number of steps taken in the text preprocessing depends on the task and the normalization steps can vary. In brand name extraction, text preprocessing may interfere with the task. For example, if a brand name contains digits, for example Q50, then a preprocessing may remove the digits. In addition, stemming may change the word, for instance "Mercedes" would change to "Merced" with Porter stemmer.

Converting text to lower case or removing the punctuations may result in the loss of useful information about some of the brand names.

The primary problem with extracting brand names from texts is ambiguity. "George", "Apple", "Bell", and "Gap" are some of the examples that can be both brand names and common nouns. In addition, words such as "Jet Ski", "Jacuzzi", and "Kleenex" were once brand names and nowadays are often used as a general word in everyday vocabulary. Hence, disambiguation is a major area of interest within the field of NER and resolving it can increase the performance of the system significantly.

On the other hand, brand names can be from different domains. For example, "Apple" is from technology domain, "Honda" is from automotive domain, and "Zara" is from fashion domain. The vocabulary that customers and companies use to write about a product in a specific domain is usually different than writing about a product from other domains and this can cause problem in NLP tasks.

In the next chapter, a proposed methodology for extracting brand names is provided.

2.8 Summary

This chapter began by describing NER, its applications, and challenges. It went on to describe representation learning in NLP. It was mentioned that bag-of-words and n-gram, as two examples of traditional document representations, are at a disadvantage of curse of dimensionality and lack of global knowledge. Therefore, word embedding approaches were discussed and Word2Vec as a method that is used in this study was introduced. Moreover, a summary of machine learning methods in NER and evaluation metrics were provided.

Furthermore, a brand name can be represented with a proper noun. Therefore, brand names are named entities and brand name extraction is a subset of Named Entity Extraction. On the other hand, the text data that contains brand names can be from different domains. This raises questions about the methods that can be used for brand name extraction which will be discussed in the next chapter.

Chapter 3

The Proposed Approach for Extracting Brand Names

3.1 Introduction

Extracting brand names is an important component in business intelligence, information extraction, creation of brands catalogue, and public opinion tracking. A brand name is a unique and original proper noun; therefore, brand name extraction can be considered as a subset of NER.

Similar to NER, brand name extraction faces many challenges. First, companies try to pick a novel and unique name since they want their brand seat in customer's mind. Therefore, the brands vocabulary is expanding daily and there is no rules, patterns or an up-to-date catalogue that can identify them in text. Second, most of the brand names are words that can also be used as common nouns and conversely, there are everyday words that once were brand and product names. As a result, ambiguity is a serious challenge in extracting brand names and it may be resolved by taking the context into consideration.

As mentioned in section 2.5, traditional machine learning can support supervised, semi-supervised and unsupervised learning. In all the mentioned learning methodologies, training and test data are taken from the same domain. However, labeling training data for brand names is expensive, time consuming and a domain expert is needed for each domain. Therefore, the domain adaptation technique can be employed in order to overcome this deficiency.

Word embedding, as discussed in section 2.4.2, has the ability to capture the meaning and the context of a word in a document, the relationship between words, and the semantic and syntactic similarity. Therefore, employing word embedding might help with resolving the ambiguity in brand name extraction when the brand names are from different domains.

In particular, this thesis will examine two main research questions:

1. Does using the context of the words improve the performance of the brand name extraction system?

2. Does using the trained word embedding instead of pre-trained models increase the performance and resolve ambiguity?

This chapter describes and discusses the methods used in this study. The first section describes the domain adaptation technique. The second section moves on to describe the proposed methodology to extract the brand names from texts in different domains using trained Word2Vec model and iterative domain adaptation.

3.2 Domain adaptation

In traditional machine learning methods, training data and test data are from the same domain (same feature space and data distribution). With the rise of big data

and the considerable data that is generated everyday in different shapes and domains, training the learners that can be applied to the data from different domains is crucial. This learning framework is called transfer learning. More recent attention has focused on the provision of transfer learning since in real-world problems, data-labeling is expensive, time-consuming, or difficult. In transfer learning, the sample space, label space, and distribution can be different in source domain and target domain. In domain adaptation, as one of the settings of transfer learning, only the probability distribution may vary where the sample and label space is the same in both domains [41].

3.2.1 Notations

In this section, some notations are introduced to facilitate discussion. These notations have been used by Pan and Weiss [42, 43] in their survey paper.

In this thesis, each domain $\mathcal{D}$ contains a feature space $\mathcal{X}$ and a marginal probability distribution P(X), where $X = \{x_1, x_2, ..., x_n\} \in \mathcal{X}$. Furthermore, for each domain $\mathcal{D}$, a task $\mathcal{T}$ contains a label space $\mathcal{Y}$, and a predictive function $f(.)$ is defined. $f(.)$ is formed using training data, pairs $\{x_i, y_i\}$ where $x_i \in X$ and $y_i \in \mathcal{Y}$, and the feature vector. Given an instance x, the label of x, which is $f(x)$, can be predicted using the function $f(.)$.

Accordingly, $\mathcal{D}_S = \{(x_{S_1}, y_{S_1}), ..., (x_{S_n}, y_{S_n})\}$ is the source domain where the data instance $x_{S_i} \in \mathcal{X}_S$ from D_S has the class label of $y_{S_i} \in \mathcal{Y}_S$. Likewise, $\mathcal{D}_T = \{(x_{T_1}, y_{T_1}), ..., (x_{T_n}, y_{T_n})\}$ is the target domain where the data instance $x_{T_i} \in \mathcal{X}_T$ from D_T has the class label of $y_{T_i} \in \mathcal{Y}_T$. The source task $\mathcal{T}_S$ and the target task $\mathcal{T}_T$ are also defined for the further references.

3.2.2 Definitions

Given a source domain $\mathcal{D}_S$ and its task $\mathcal{T}_S$, a target domain $\mathcal{D}_T$ and its task $\mathcal{T}_T$, transfer learning is based on the idea of improving target predictive function $f_T(.)$ in $\mathcal{D}_T$ by employing the information and knowledge from $\mathcal{D}_S$ and $\mathcal{T}_S$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$.

Domain adaptation, also known as transductive transfer learning, happens when $\mathcal{T}_S$ and $\mathcal{T}_T$ are the same, but $\mathcal{D}_S$ and $\mathcal{D}_T$ are different. In this case, all the knowledge is from the source domain with a great amount of labeled data where the target domain has no labeled data.

In domain adaptation, unlike the marginal probability the feature space in source and target domains are the same. Therefore, $\mathcal{X}_S = \mathcal{X}_T$ and $P(X_S) \neq P(X_T)$ [42].

3.2.3 Iterative Training in Domain Adaptation

Using domain adaptation, the knowledge from labeled data in source domain is employed to gradually label instances from the target domain which has none or a small amount of labeled data. There are different types of problems in which the iterative training can be applied [44]:

- **The buildup of a new corpus using iterative learning:** A model is trained using labeled source data x_{S_i}. This model is then employed to classify x_{T_i} and add the data with the highest confidence score to the source data based on a fixed threshold. This process will be repeated several times and each time the number of labeled data from the target domain will increase in the training corpus.

- **Learning using multiple source datasets:** In this setting, some or all the source domains are joined together to build a training corpus.

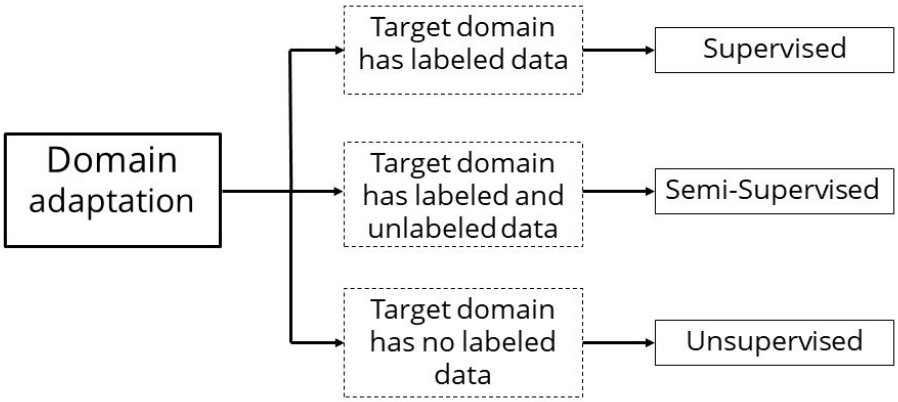


Figure 3.1: Different types of domain adaptation based on target labels

- **Making a decision based on the vote of each source's classifier:** The idea is to build a models for each source corpus and the final classifier will be selected based on the decision of all the classifiers.

- **The combination of voting and iterative learning:** It is based on training N models from N source domains. The data with the highest confidence score then will be added to the training dataset.

The number of iterations is based on the nature of the problem. The batch domain adaptation is when the iteration happens only once. Moreover, domain adaptation is categorized into supervised, semi-supervised and unsupervised which is based on the status of labeled data in target domain (Fig. 3.1) [45]. In supervised domain adaptation, the target domain has labeled data. In semi-supervised domain adaptation, the target domain has both labeled and unlabeled data, and there is no labeled data in unsupervised domain adaptation.

3.3 The Proposed Approach

In this section, the proposed approach for extracting brand names has been discussed and is shown in Fig. 3.2. First, a Word2Vec model is trained using the forum posts from different domains. Second, a feature set is created using the context window idea and the trained word embedding model. A domain adaptation technique is then applied to build a learning model for predicting brand names in other domains and finally, the system is evaluated using evaluation metrics.

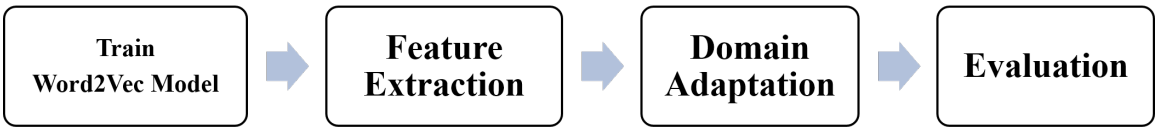


Figure 3.2: The process of proposed approach.

The results are based on textual data in English language provided by VerticalScope company[1], Toronto. As an integrated multi-platform media company founded in 1999, VerticalScope has an enormous textual dataset gathered from more than 800 websites with more than 25 million aggregate pages of content and more than 125 Million unique visitors per month [46]. The provided dataset contained information about forum posts written in different verticals (domains), such as automotive, powersports, outdoor, home, health, and technology. However, not all of the

[1]https://www.verticalscope.com/

domains have labeled data; automotive, powersports and a relatively small number of outdoor instances have manual human-labelled data. The high cost of generating training data for each domain made the company to seek other solutions to identify the divergent types of products from all verticals. Each dataset contains forum posts from a specific domain and each post contains one or several sentences and is accompanied by indices indicating the position of brand names in the sentences. Hence, the problem is a binary classification; true if the word is a brand name, false otherwise.

3.3.1 Training Word2Vec Model

To date various methods have been developed and introduced to generate word vectors. In this study, a CBOW Word2Vec model was trained on the text data of forum posts in different domains. For training, 20,567,045 sentences were feed to the Word2Vec constructor in the *gensim* Python library.

The attributes in Word2Vec constructor in the *gensim* are defined as follows:

- **size:** The dimension of the word vectors.
- **window:** the maximum distance between the target word and its surrounding words.
- **min_count**: The model would ignore the words that have the frequency less than this value. In large datasets, the extremely infrequent words are mainly frivolous.
- **sg:** It specifies the training algorithms. CBOW or skip-gram.

In this study, the size parameter is set to 300 which is a default size in most word representation problems. Window is set to 5 which means any neighbour which is

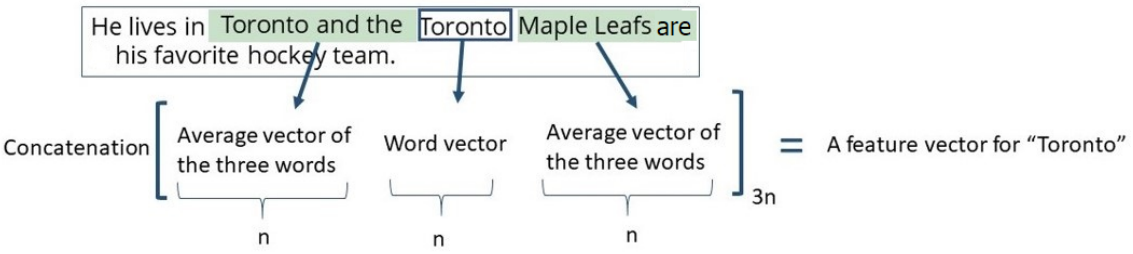


Figure 3.3: The feature extraction for word "Toronto". The context window of size 7 and a feature set size of $3n$

located in the distance more than 5, is not treated as a related word of the target word. In addition, min_count is set to 5, and sg to CBOW.

3.3.2 Feature Extraction

In the Word2Vec model which was described in section 3.3.1, each word is mapped to a vector. In this study, each word was represented by its surrounding words. Hence, a context window containing k consecutive words , $k/2$ words before and $k/2$ words after the current word, was considered as a representation for each word. Each word in the context window can be depicted by a corresponding vector from the Word2Vec model. Given the context window, the concatenation of the average vector of the words before the current word, the vector of the current word, and the average vector of the words after the current word is set as the word feature. If the dimensionality of the word vectors in the Word2Vec model was n, the result vector of the context window would be $3n$, which is the dimensionality of the features set. An overview of the feature extraction is shown in Fig. 3.3.

For the words at the beginning or end of the posts, zero vector is considered for missing words. In addition, zero padding technique is used for OOV words.

3.3.3 Domain Adaptation

In the available dataset, the limited number of annotation in some domains was evident. Since automotive domain had a considerable amount of labeled data, an unsupervised iterative domain adaptation was employed to resolve the lack of labels in other domains. While $\mathcal{D}_S$ and $\mathcal{D}_T$ are different in this problem, the $\mathcal{T}_S$ and $\mathcal{T}_T$ are both extracting the brand names. Furthermore, the feature space $\mathcal{X}_S = \mathcal{X}_T$ and they were represented as it was described in section 3.3.2.

In this study, the iterative training in domain adaptation was used (section 3.2.3). As shown in Figure 3.4, the labeled data from a source domain are taken to gradually label instances from the target domain which has none or a small amount of labeled data. In each iteration, the data with the higher confidence score is added to the training dataset using the model that was trained on the labeled source data. The same amount of added data is randomly removed from the training data. This process is repeated several times and each time the number of labeled data from the target domain is increased in the training corpus.

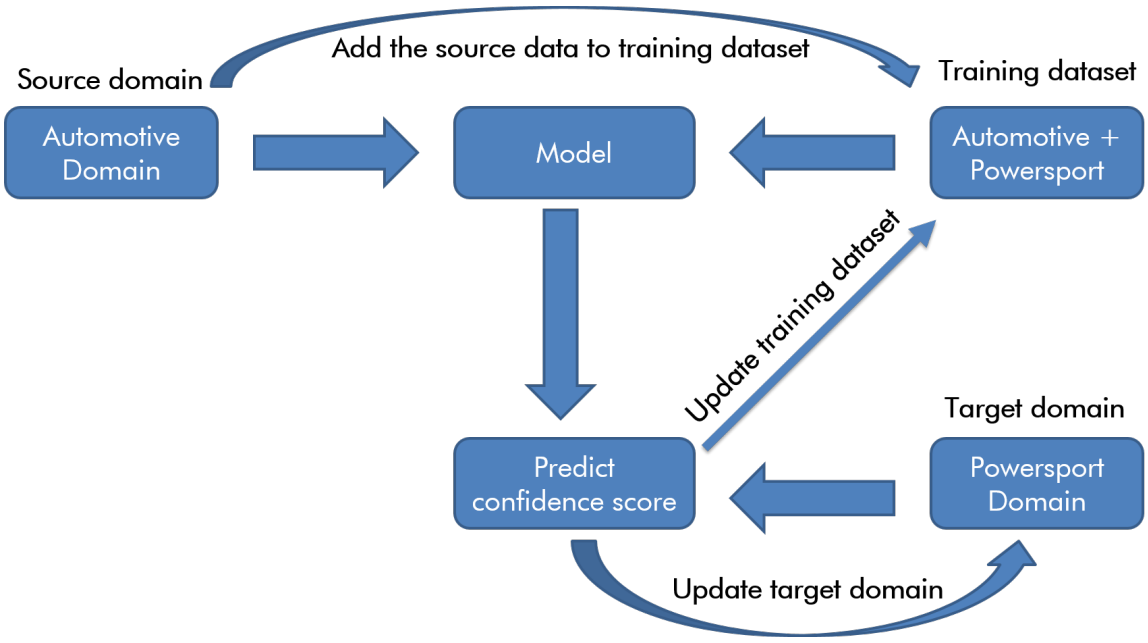


Figure 3.4: Domain adaptation process when the source domain is automotive and target domain is powersport

3.4 Summary

Two main research questions were raised in the beginning of this chapter:

1. Does using the context of the words improve the performance of the brand name extraction system?

2. Does using the trained word embedding instead of pre-trained models increase the performance and resolve ambiguity?

The rest of the chapter explained domain adaptation technique in details. In contrast to traditional machine learning methods, domain adaptation is used when the training and test datasets are from different domains.

The proposed approach, which was explained in the last section, employed Word2Vec model and context window to represent the words. This representation was used in a

supervised learning algorithm during domain adaptation process. The domain adaptation procedure and the results obtained from it are described in the next chapter.

Chapter 4

Experiments, results, and discussion

4.1 introduction

In this chapter, different experiments that were done in this study are discussed. There were different settings for the proposed approach that needed to be surveyed: the properties of the context window; the type of word embedding model; and the learning algorithm used in the domain adaptation.

All the experiments were done on word level and, therefore, the sentences were tokenized and each words was given the specified label. As mentioned in the previous chapter, text preprocessing may interfere with the brand names extraction. For this reason, no specific preprocessing was done on the data.

Furthermore, the ideal scenario in every business problem is to achieve the maximum precision and recall. However, in reality Machine Learning (ML) technologies have their own deficiencies which prevents the business owners from having the best of two worlds. For example, in selecting the potential customers for a certain product,

it is more important to catch every possible customer even if it means the marketing department might need to call some false positive. In this research, the evaluation was based on the recall, since the recall metric is more important in the industry; companies tend to lose less positive data and have a prediction with higher coverage on the required data.

In the experiments of this chapter, the evaluations are the average performance of ten runs of the approach on a randomly shuffled dataset. In addition, the number of instances in automotive domain is 80,000, the powersport domain 7,000, and the outdoor domain 6,775.

4.2 Context Window

The idea of using a context window was discussed in section 3.3.2. Context window can capture the word context and, therefore, may reduce the ambiguity. In this section, three main questions about the context window are surveyed regarding recall:

1. What is the best size for the context window?
2. Should the context window include the main word or not?
3. Should the context window contain the vectors of the surrounding words or the average of the vectors?

4.2.1 The Performance Of using Average Vectors and Actual Vectors

The context window contains the vectors of the word and its surroundings. When the context window's size is greater than 1, more than one vector would be in the

Table 4.1: Training dataset Automotive - Test dataset Automotive - The main word excluded in the context window

Context Window Size	**Precision**	**Recall**	**F-score**
3	0.42	0.14	0.21
5	0.70	0.05	0.1
7	0.47	0.01	0.02
9	0.25	0	0
11	0.47	0.02	0.04
13	0	0	0
15	0	0	0

window. Hence, considering the average vector of these vectors may result in a better performance.

Table 4.2 provides the results obtained from using the actual vector instead of the average when the training set and test set are from the same domain and similarly Table 4.3 provides the results when the source and target domains are different. Since the dimensionality of each word vectors is 300, the feature size for one word in the context window of size 3, is 900. Likewise, the feature size of a word in the context window of size 5 is 1500 and with the context window of size 7 is 2100. In this scenario, it wasn't feasible to examine the higher size of context window since the feature set size was increasing linearly. The performance of the learning with the average vectors is provided in the next section. Taken together, these results suggest that using the average vector improve the performance and decrease the amount of memory needed for the feature set significantly.

Table 4.2: Training dataset Automotive - Test dataset Automotive - The context window contain the actual vectors

Context Window Size	Precision	Recall	F-score
3	0.81	0.75	0.78
5	0.67	0.82	0.73
7	0.76	0.69	0.73

Table 4.3: Training dataset Automotive - Test dataset PowerSport - The context window contain the actual vectors

Context Window Size	Precision	Recall	F-score
3	0.80	0.58	0.67
5	0.63	0.67	0.65
7	0.73	0.53	0.62

4.2.2 Context Window with the Average Vectors

Changes in the recall and precision were compared using different sizes of context window. The context window with the length of five, seven, nine, and eleven were used in this study. The differences between the performance metrics using different context window size are highlighted in Table 4.4. In this table, the training and test datasets are from the same domain and Linear SVM learning algorithm is used for classification. Table 4.5 also provides the results obtained from using two different domains as training and test datasets. Other settings remained unchanged for a better comparison.

The results of the Table 4.4 and Table 4.5 are plotted in Figure 4.1 and Figure 4.2 respectively to show the non-dominated (Pareto-front) points. As it was mentioned before, the aim of these experiments is to maximize the recall measurement. However,

in the batch domain adaptation in which the findings is used in iterative domain adaptation, the aim was to maximize both recall and precision. Therefore, a Pareto front curve was employed to find the points which maximize both objective function parameters; on the Pareto front curve, a set of optimum solution can be found in a way that no better and feasible solutions can be found [47].

It is apparent from this figure that context window of size seven has a better performance regarding precision and recall compared to other window lengths.

Table 4.4: Training dataset Automotive - Test dataset Automotive

Context Window Size	**Precision**	**Recall**	**F-score**
3	0.81	0.76	0.78
5	0.72	0.83	0.77
7	0.84	0.76	0.79
9	0.83	0.73	0.78
11	0.84	0.71	0.77
13	0.90	0.60	0.72
15	0.85	0.63	0.72

Table 4.5: Training dataset Automotive - Test dataset Powersport

Context Window Size	**Precision**	**Recall**	**F-score**
3	0.78	0.55	0.64
5	0.76	0.57	0.65
7	0.80	0.62	0.69
9	0.81	0.53	0.64
11	0.80	0.48	0.60
13	0.84	0.42	0.56
15	0.78	0.43	0.55

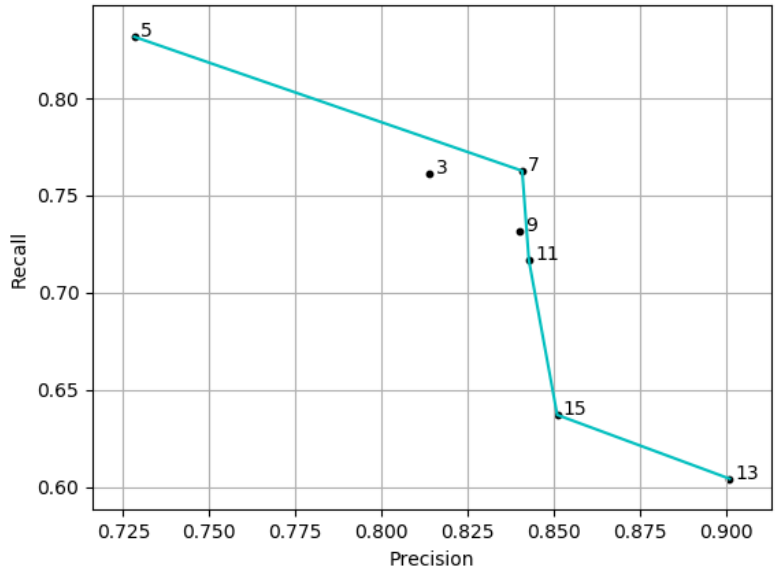


Figure 4.1: The comparison between the context window size when the training dataset and test dataset are taken from the same domain.The line shows the non-dominated (Pareto-front) points. Model: locally trained Word2Vec.

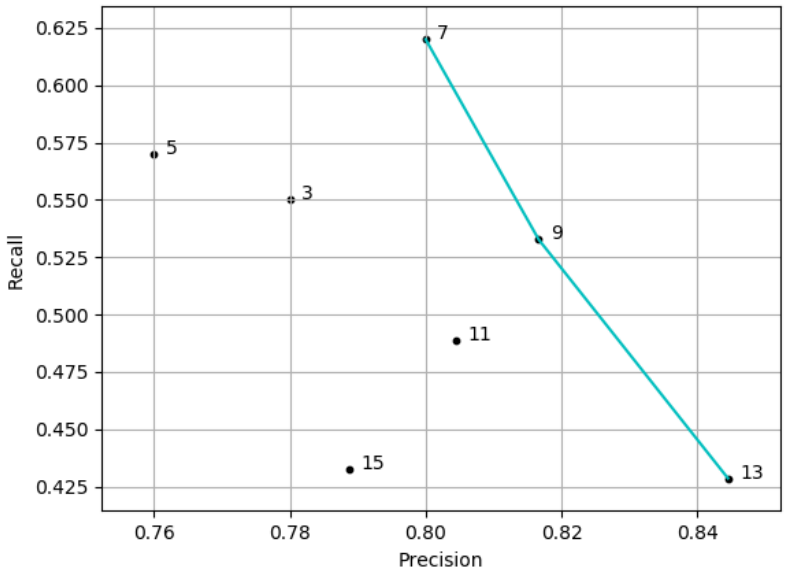


Figure 4.2: The comparison between context window size when the training dataset and test dataset are taken from different domains. The line shows the non-dominated (Pareto-front) points. Model: locally trained Word2Vec

4.2.3 The Performance of Inclusion and Exclusion of the Main Word

There are two settings that can be employed for making the context window: 1. The context window with the current word included 2. The context window contains the surrounding words and not the current word.

The results for different sizes of context window excluding the main word are shown in Table 4.1. In this example, the training set and test set are taken from the same domain. Comparing the result in Table 4.1 and Table 4.4, it can be seen that the performance of the scenario with the inclusion of the main word is higher than the other one.

4.2.4 Discussion

As can be seen from the Table 4.4, the proposed algorithm offers a very low recall when the main word is excluded from the context window. This result was expected since removing the main word would remove the meaning and context of the word from the feature set.

Moreover, comparing the Table 4.2 and the Table 4.4, the model shows a lower performance when the actual vectors are used instead of the average vector. This result may be explained by the fact that the order of the words loses its importance when the average vector is used where the average vector depict the context of the surrounding words. Furthermore, using the actual vector is not feasible for the larger size of context windows since the feature set size would be increased linearly.

Finally, as can be seen in Fig. 4.1 and Fig. 4.2, the best performance, both when the source and target domains are the same and when they are different, is when the context window is of size 7. A possible explanation for this results may be the

average length of English sentences. When the size of the context window increases, the connection between the words may decrease.

Taken together, these results suggest that the model might get a better recall when the size of the context window is 7, the main word included in the context window, and the average vectors are used for surrounding words.

4.3 Word2Vec Model

Google released a pre-trained vectors contains 3 million words and phrases in 2013 [48]. This model was trained on roughly 100 billion words from Google News dataset and has 300-dimensional vectors [1].

Furthermore, as mentioned in chapter 3 section 3.3.1, a Word2Vec model can be trained using texts per application. A case study approach was used in [49] to examine the performance of an NER system when using locally trained Word2Vec. In this study, a Word2Vec model was trained on texts from forum posts in different domains. The Word2Vec constructor parameters was discussed in the mentioned section. The values that was employed in this study was the window size of 5, the vector dimensionality of 300, and the min_count of 5. The dimensionality was chosen the same as Google's pre-trained vectors for a fair comparison.

Figure 4.3 and Figure 4.4 shows the Precision-Recall performance of Google's pre-trained Word2Vec and locally trained model with two different context window size. Table 4.6 and Table 4.7 also presents the detailed results of this comparison. As can be seen from the tables, the locally trained model performs better specially when the training and test data are from different domains.

[1] https://drive.google.com/file/d/0B7XkCwpI5KDYNlNUTTlSS21pQmM/edit

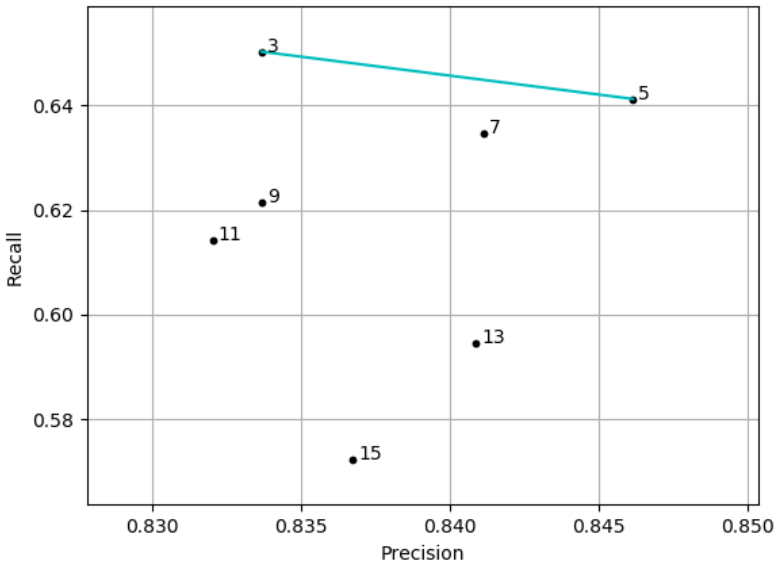


Figure 4.3: The comparison between context window size when the training dataset and test dataset are taken from the same domain. The line shows the non-dominated (Pareto-front) points. Model: Google's pre-trained Word2Vec

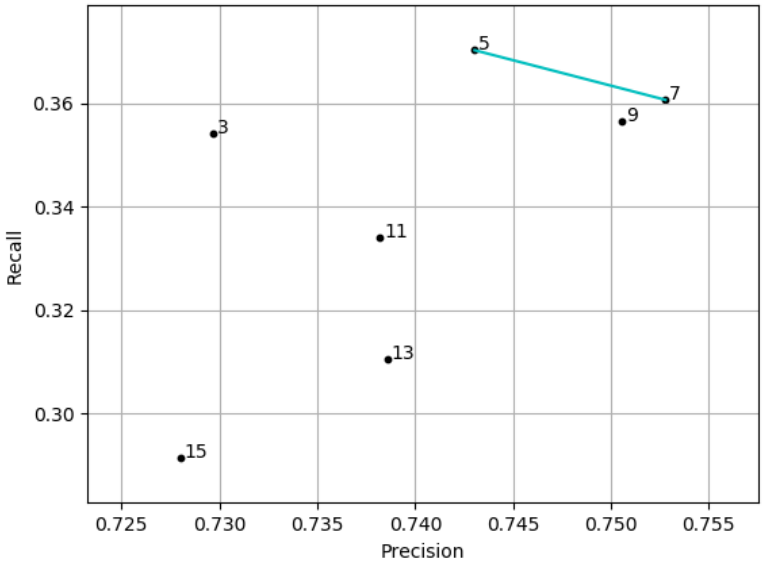


Figure 4.4: The comparison between context window size when the training dataset and test dataset are taken from different domains. The line shows the non-dominated (Pareto-front) points. Model: Google's pre-trained Word2Vec

Table 4.6: Training dataset Automotive - Test dataset Automotive - Model: Google's pre-trained Word2Vec

Context Window Size	Precision	Recall	F-score
3	0.83	0.65	0.73
5	0.84	0.64	0.72
7	0.84	0.63	0.72
9	0.83	0.62	0.71
11	0.83	0.61	0.70
13	0.84	0.59	0.69
15	0.83	0.57	0.67

Table 4.7: Training dataset Automotive - Test dataset PowerSport - Model: Google's pre-trained Word2Vec

Context Window Size	Precision	Recall	F-score
3	0.72	0.35	0.47
5	0.74	0.37	0.49
7	0.75	0.36	0.48
9	0.75	0.35	0.48
11	0.73	0.33	0.45
13	0.73	0.31	0.43
15	0.72	0.29	0.41

4.3.1 Discussion

The results of using pre-trained Word2Vec model when the training set and test set are from the same domain was shown in 4.6. Turning now to the results of using locally trained Word2Vec model in Table 4.4, it can be seen that the performance of the latter is higher than the pre-trained model. For example, for the context window of size 7, the recall of locally trained model is %76 and the recall of pre-trained model

is %63.

The difference is more when the taining set and test set are from different domains. Comparing the two results in Table 4.5 and Table 4.7, it can be seen that for window size of 7, the recall is %55 for the locally trained Word2Vec while it is %36 for the other.

A comparison of the two results reveals that a locally trained Word2Vec performs better than a general pre-trained model. It can be seen that it is specially performs better when the training dataset and test dataset are from different domains.

Together these results provide important insights into the importance of locally trained word embedding models. The observed increase in recall could be attributed to the fact that locally trained models can capture the essence of the domain and may perform better in the domain specific tasks compared to general pre-trained models.

4.4 Learning algorithm

Different learning algorithms can be employed in the domain adaptation technique. For comparing the algorithms, one iteration of learning are used for comparison. In this study, logistic regression, SVM (with rbf kernel), and linear SVM were compared when the context window size is 7 and the word embedding model is locally trained.

As shown in Figure 4.5 and Figure 4.6, the performance of Linear SVM was more promising than the other algorithms. Detailed results are shown in Table 4.8 and Table 4.9 and confirms that when two setting of the problem, one domain and two domains, take into consideration, Linear SVM works better than logistic regression or SVM (rbf kernel).

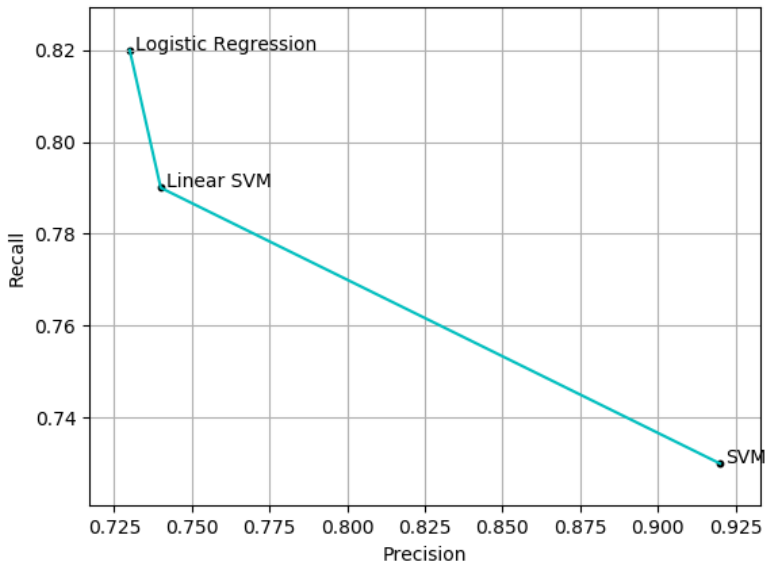


Figure 4.5: The comparison between different algorithms when the training dataset and test dataset are taken from the same domain. The line shows the non-dominated (Pareto-front) points. Model: locally trained Word2Vec

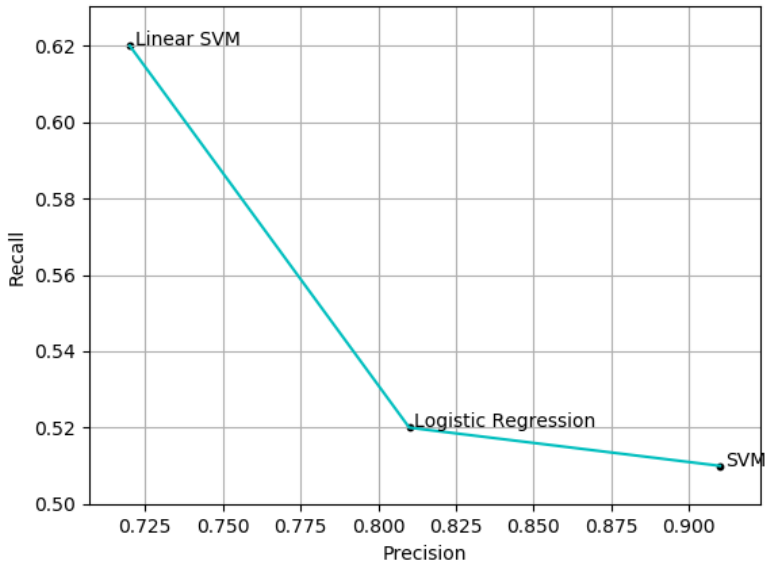


Figure 4.6: The comparison between different algorithms when the training dataset and test dataset are taken from different domains. The line shows the non-dominated (Pareto-front) points. Model: locally trained Word2Vec

Table 4.8: Training dataset Automotive - Test dataset Automotive - Different learning algorithms

Learning algorithm	**Precision**	**Recall**	**F-score**
Logistic Regression	0.73	0.82	0.77
SVM (rbf kernel)	0.92	0.73	0.81
Linear SVM	0.74	0.79	0.76

Table 4.9: Training dataset Automotive - Test dataset PowerSport - Different learning algorithms

Learning algorithm	**Precision**	**Recall**	**F-score**
Logistic Regression	0.81	0.52	0.63
SVM (rbf kernel)	0.91	0.51	0.65
Linear SVM	0.72	0.62	0.66

4.4.1 Discussion

From the Table 4.8, logistic regression algorithm showed a better recall performance compared to the two other algorithms when the source and target domain are the same. On the other hand, when the target and source domain are different, linear SVM performs better than logistic regression and SVM based on Table 4.9.

These results were expected, since a considerable number of text classification problems are linearly separable [50]. Besides, when the number of instances and features are large, like in text classification, mapping data to a higher dimensional space (nonlinear mapping) does not improve the performance [51]. In addition, the linear kernel is faster [51] and there are fewer parameters to optimize compared to other kernels.

As a result, it was decided to use linear SVM in the domain adaptation process.

4.5 Domain adaptation

There were three labeled domain specific datasets in the dataset that were used: automotive, powersport, and outdoor. When people talk about their hobbies and interests in different domains, their use of words and the general characteristics of their sentences are different. For instance, when someone is talking about his favorite car is different than when he is talking about the outdoor equipment.

Furthermore, automotive and powersport are relatively similar domains based on the way people tend to talk about them and contrarily automotive and outdoor domains have different characteristics. Therefore, the study of the performance of domain adaptation were broke down into two subsections: Domain adaptation on similar domains (experiment 1) and domain adaptation on dissimilar domains (experiment 2).

4.5.1 Experiment 1

Automotive and powersport are two domains in the available dataset with the labeled data. The examples of brand names in automotive domain are Honda, NSX, Land Rover, CRV and for the powersport domain are Stryker, Harley, and T120.

Table 4.10 presents the results obtained from domain adaptation process with the automotive as the source domain (training dataset), and powersport as the target domain (test dataset).

Table 4.10: Training dataset Automotive - Test dataset Powersport

Method	Precision	Recall	F-score
Batch domain adaptation	0.80	0.62	0.69
Iterative domain adaptation	0.84	0.71	0.77

4.5.2 Experiment 2

Automotive and outdoor are two domains in the available datasets. In addition, the labeled dataset for outdoor domain was relatively small. The examples of brand names in automotive domain are Senna, Malibu, E30 and for the outdoor domain are Three Rivers Archery, Hamskea, and Nature-Trek, and Ruger.

The results obtained from domain adaptation with automotive as the source domain and outdoor as the target domain can be compared in Table 4.11. It is apparent from this table that there was a significant difference between the two settings. In batch domain adaptation, the recall of 0.06 was achieved which was expected since the two domains are completely different. However, a comparison of the results between batch domain adaptation and iterative domain adaptation reveals that adding the instances from outdoor domain with the higher confidence score to the source data improved the performance of the classifier significantly. In each iteration, a predefined percentage of instances from target domain with the highest confidence score is added to the training set; by the end of the iterations, half of the main target domain was transferred to the training set.

Furthermore, in Figure 4.7 there is a clear trend of decreasing in F-score as the iterations continued. The F-score in this table is the F-score of the data from the target domain that were added to the training set up to the specified iteration.

Table 4.11: Training dataset Automotive - Test dataset Outdoor

Method	Precision	Recall	F-score
Batch domain adaptation	0.75	0.06	0.11
Iterative domain adaptation	0.46	0.58	0.51

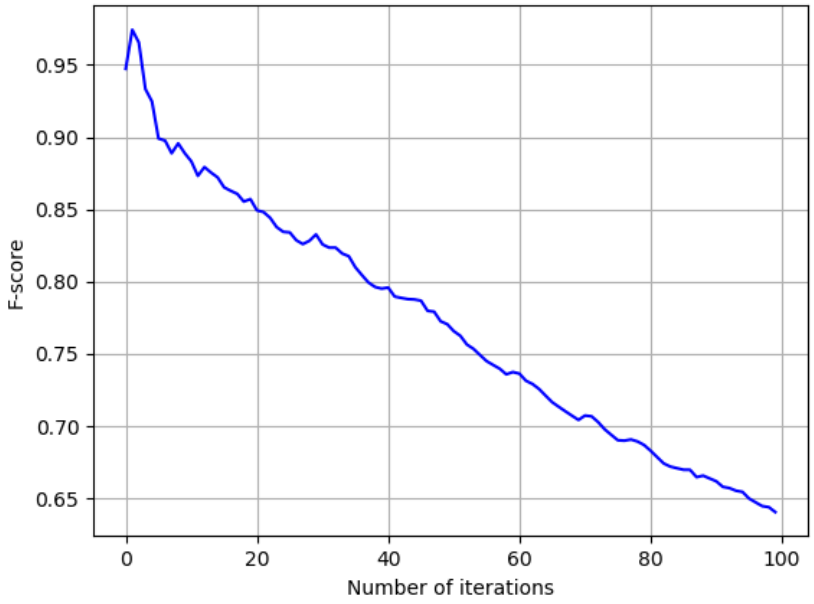


Figure 4.7: F-score decreases as more iterations are performed.

Discussion

This study set out with the aim of transferring the knowledge from a domain with labeled data to a new domain. The procedure was successful as it was able to increase the recall in two new domains: powersport, and outdoor.

First, the recall metric of batch domain adaptation when the powersport is the target domain was %62 (Table 4.10). It seems possible that these results are due to the similarity of automotive and powersport domain. However, the iterative domain adaptation increased the recall by %9 which can be explained by the added data from

the powersport domain to the training set.

Second, automotive and outdoor domains are dissimilar domains based on the nature of the brand names and also the words that users employ to write about their experience. This claim can be confirmed by the recall metric of batch domain adaptation shown in Table 4.11. However, the recall increased by %52 with performing the iterative domain adaptation.

Finally, the decrease of the F-score was illustrated in Figure 4.7. It is difficult to explain this result, since in each iteration, the training set contains more and more data from the outdoor domain. However, there are several possible explanations for this result. First, as the iteration continues, the information in the target domain gets more domain specific. Second, although the training set contains data from the outdoor domain, it is still relatively low compared to the number of instances from the automotive domain. Therefore, even though the recall stays high during the iterations, precision drops and caused the decrease in the F-score.

4.6 Summary

The results in this Chapter indicate that domain adaptation can increase the performance of brand name extraction system significantly. Firstly, different experiments were designed to choose the best settings that need to be used in domain adaptation procedure; the results shows the optimal context window's properties, Word2Vec model, and learning algorithm that can increase the performance of the system.. Then, these settings were used in the domain adaptation procedure for two different target domains. The %9 recall improvement when the target domain is powersport and %52 recall improvement when the target domain is outdoor was achieved in these experiments.

These results answered the two research questions in the previous Chapter. Using word context improved the performance of the system. Moreover, the locally trained Word2Vec performed better compared to the pre-trained Word2Vec.

Chapter 5

Conclusions and future works

5.1 Summary and Conclusions

Extracting brand names plays an important role in business intelligence, information extraction, and the creation of brand catalogues. In addition, labeling training data for brand names is expensive, time-consuming and a domain expert is needed for each domain. This thesis was undertaken to design a brand name extraction system and evaluate the performance of using locally trained word embedding in the design.

The results in this thesis were based upon textual data in English language provided by VerticalScope company, Toronto. The dataset contains forum posts in different domains with labeled brand names; it belongs to VerticalScope company and can not be shared. However, the proposed method is independent of the dataset and can be applied on other textual datasets with similar specifications.

Returning to the questions posed at the beginning of Chapter 3, it is now possible to state that first, using the context of the words improved the performance of brand name extraction. Second, the locally trained Word2Vec performed better compared to the pre-trained Word2vec; it improved the recall by %13 when the source and

target domains are the same and by %19 when they are different.

Moreover, a domain adaptation technique was employed since in the traditional machine learning approaches, the training and test data are taken from the same domain. The experiments have shown that domain adaptation on similar source and target domains, such as automotive and powersport, increased the recall by %9 . In addition, the domain adaptation on dissimilar source and target domains, such as automotive and outdoor, increased the recall by %52.

This research has several practical applications. Firstly, it points to the advantage of using locally trained word embedding in NER. This understanding can be employed in other NLP tasks. Secondly, in the NLP world, where the labeled data is valuable, using the domain adaptation techniques can be advantageous. Although the current study is based on a small number of domains, the findings suggest that the proposed method can be used and improved in other similar problems.

5.2 Limitations and Future works

The current investigation was limited by the lack of labeled data in different domains. Therefore, with having more labeled data, one can create a new source dataset using a mixture of different domains and examine the performance.

There are many ways in which this work can be continued.

Resolving the named entities ambiguity problem As it was discussed in section 2.3, named entities ambiguity is one of the biggest challenges in NER. There is no method that can be claimed to find the problematic words completely. In this study the locally trained word embedding partially resolved the ambiguity; however, a further study could add other techniques to the study for disambiguation.

Working on Out of Vocabulary In section 2.4.2, the approaches to handle OOV was explained. In this study, zero padding was employed for the words that are not found in the word embedding model. Further research in this field would be of great help in word embedding studies.

Examining the proposed method for extracting other types of named entities In the majority of NLP studies, finding an authentic labeled dataset is one of the first issues a researcher faces. In this study, three datasets from different domains with brand names label was used. Further research on different datasets might investigate the performance of the proposed system on extracting other types of named entities.

Bibliography

[1] Benjamin Strauss, Bethany Toma, Alan Ritter, Marie-Catherine De Marneffe, and Wei Xu. Results of the wnut16 named entity recognition shared task. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 138–144, 2016.

[2] Lisa F Rau. Extracting company names from text. In *[1991] Proceedings. The Seventh IEEE Conference on Artificial Intelligence Application*, volume 1, pages 29–32. IEEE, 1991.

[3] Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2145–2158, 2018.

[4] Rohini Srihari and Wei Li. Information extraction supported question answering. Technical report, CYMFONY NET INC WILLIAMSVILLE NY, 1999.

[5] Diego Mollá, Menno Van Zaanen, Daniel Smith, et al. Named entity recognition for question answering. 2006.

[6] Julia Hirschberg and Christopher D Manning. Advances in natural language processing. *Science*, 349(6245):261–266, 2015.

[7] Bogdan Babych and Anthony Hartley. Improving machine translation quality with automatic named entity recognition. In *Proceedings of the 7th International EAMT workshop on MT and other Language Technology Tools, Improving MT through other Language Technology Tools: Resources and Tools for Building MT*, pages 1–8. Association for Computational Linguistics, 2003.

[8] Yufeng Chen, Chengqing Zong, and Keh-Yih Su. A joint model to identify and align bilingual named entities. *Computational linguistics*, 39(2):229–266, 2013.

[9] Lei Zhang and Bing Liu. Aspect and entity extraction for opinion mining. In *Data mining and knowledge discovery for big data*, pages 1–40. Springer, 2014.

[10] Kumar Ravi and Vadlamani Ravi. A survey on opinion mining and sentiment analysis: tasks, approaches and applications. *Knowledge-Based Systems*, 89:14–46, 2015.

[11] Jason J Jung. Online named entity recognition method for microtexts in social networking services: A case study of twitter. *Expert Systems with Applications*, 39(9):8066–8070, 2012.

[12] Damiano Spina, Julio Gonzalo, and Enrique Amigó. Discovering filter keywords for company name disambiguation in twitter. *Expert Systems with Applications*, 40(12):4986–5003, 2013.

[13] Eva Blomqvist. The use of semantic web technologies for decision support–a survey. *Semantic Web*, 5(3):177–201, 2014.

[14] Ivan Habernal and Miloslav KonopíK. Swsnl: semantic web search using natural language. *Expert Systems with Applications*, 40(9):3649–3664, 2013.

[15] Jenny Rose Finkel and Christopher D Manning. Nested named entity recognition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 141–150. Association for Computational Linguistics, 2009.

[16] Archana Goyal, Vishal Gupta, and Manish Kumar. Recent named entity recognition and classification techniques: A systematic review. *Computer Science Review*, 29:21–43, 2018.

[17] Sara Keretna, Chee Peng Lim, Doug Creighton, and Khaled Bashir Shaban. Enhancing medical named entity recognition with an extended segment representation technique. *Computer methods and programs in biomedicine*, 119(2):88–100, 2015.

[18] Yotam Eshel, Noam Cohen, Kira Radinsky, Shaul Markovitch, Ikuya Yamada, and Omer Levy. Named entity disambiguation for noisy text. *arXiv preprint arXiv:1706.09147*, 2017.

[19] Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 359–367. Association for Computational Linguistics, 2011.

[20] Xiaohua Liu, Furu Wei, Shaodian Zhang, and Ming Zhou. Named entity recognition for tweets. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 4(1):3, 2013.

[21] Rahul Sharnagat. Named entity recognition: A literature survey. *Center For Indian Language Technology*, 2014.

[22] Julian SzymańSki. Comparative analysis of text representation methods using classification. *Cybernetics and Systems*, 45(2):180–199, 2014.

[23] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[24] Geoffrey E Hinton et al. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*, volume 1, page 12. Amherst, MA, 1986.

[25] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. Evaluation methods for unsupervised word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 298–307, 2015.

[26] Mohsen Mesgar and Michael Strube. Lexical coherence graph modeling using word embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1414–1423, 2016.

[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[28] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[29] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[30] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[31] Sudha Morwal, Nusrat Jahan, and Deepti Chopra. Named entity recognition using hidden markov model (hmm). *International Journal on Natural Language Computing (IJNLC)*, 1(4):15–23, 2012.

[32] Mahathi Bhagavatula, Santosh GSK, and Vasudeva Varma. Named entity recognition an aid to improve multilingual entity filling in language-independent approach. In *Proceedings of the first workshop on Information and knowledge management for developing region*, pages 3–10. ACM, 2012.

[33] Zhiqiang Toh, Bin Chen, and Jian Su. Improving twitter named entity recognition using word representations. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 141–145, 2015.

[34] S Thenmalar, J Balaji, and TV Geetha. Semi-supervised bootstrapping approach for named entity recognition. *arXiv preprint arXiv:1511.06833*, 2015.

[35] Zornitsa Kozareva, Boyan Bonev, and Andres Montoyo. Self-training and co-training applied to spanish named entity recognition. In *Mexican International conference on Artificial Intelligence*, pages 770–779. Springer, 2005.

[36] Tsendsuren Munkhdalai, Meijing Li, Taewook Kim, Oyun-Erdene Namsrai, Seon-phil Jeong, Jungpil Shin, and Keun Ho Ryu. Bio named entity recognition based on co-training algorithm. In *2012 26th International Conference on Advanced Information Networking and Applications Workshops*, pages 857–862. IEEE, 2012.

[37] Atefeh Zafarian, Ali Rokni, Shahram Khadivi, and Sonia Ghiasifard. Semi-supervised learning for named entity recognition using weakly labeled training data. In *2015 The International Symposium on Artificial Intelligence and Signal Processing (AISP)*, pages 129–135. IEEE, 2015.

[38] Scott Spangler, Ying Chen, Larry Proctor, Ana Lelescu, Amit Behal, Bin He, Thomas D Griffin, Anna Liu, Brad Wade, and Trevor Davis. Cobra–mining web for corporate brand and reputation analysis. *Web Intelligence and Agent Systems: An International Journal*, 7(3):243–254, 2009.

[39] Mohamed M Mostafa. More than words: Social networks' text mining for consumer brand sentiments. *Expert Systems with Applications*, 40(10):4241–4251, 2013.

[40] Robert Malouf, Bradley Davidson, and Ashli Sherman. Mining web text for brand associations. In *AAAI spring symposium: computational approaches to analyzing weblogs*, pages 125–127, 2006.

[41] Wouter M Kouw. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

[42] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[43] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):9, 2016.

[44] Anne Garcia-Fernandez, Olivier Ferret, and Marco Dinarelli. Evaluation of different strategies for domain adaptation in opinion mining. In *Language Resources Evaluation Conference (LREC)*, 2014.

[45] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018.

[46] VerticalScope Inc. Our history, 2019.

[47] Yukihiro Murata, Mitsushi Abe, and Ryuya Ando. Design optimization for superconducting bending magnets using pareto front curve. In *Journal of Physics: Conference Series*, volume 897, page 012023. IOP Publishing, 2017.

[48] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[49] Scharolta Katharina Sienčnik. Adapting word2vec to named entity recognition. In *Proceedings of the 20th nordic conference of computational linguistics, nodalida 2015, may 11-13, 2015, vilnius, lithuania*, number 109, pages 239–243. Linköping University Electronic Press, 2015.

[50] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[51] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.