DUPLICATE DETECTION IN MULTI-DOMAIN COMMUNITY QUESTION ANSWERING

K.K.Rasika Kariyawasam

168233K

Degree of Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

April 2020

DUPLICATE DETECTION IN MULTI-DOMAIN COMMUNITY QUESTION ANSWERING

K.K.Rasika Kariyawasam

168233K

Dissertation submitted in partial fulfillment of the requirements for the degree Master of Science in Computer Science specializing in Data Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

April 2020

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief, it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my dissertation, in whole or in part, in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature: …… Date:…….

Name: K.K.Rasika Kariyawasam

I certify that the declaration above by the candidate is true to the best of my knowledge and that this report is acceptable for evaluation for the CS6997 MSc Research Project.

Signature of the supervisor: …… Date:……..

Name: Dr. Surangika Ranathunga

Abstract

Community based question answering forums are very popular these days. People tend to refer community forums for opinions in various fields such as electronics, medical and automobile. It is very easy and useful to find a good opinion freely, but it is hard to choose the correct one when there are thousands of reviews.

There have been several efforts to automate the activities of community-based question answering systems, such as the selection of the most relevant answers to the question (question comment similarity), and identifying the questions already posted that are similar to the new question (question-question similarity). However, there are fewer attempts taken to automate the process of duplicate detection in community question answering systems. At the moment, it is the community itself that manually detects duplicates. The automation attempts are more into individual domains.

The objective of this research is to implement a mechanism that effectively identifies duplicate questions in a data set consisting of question-answer sets from multiple domains. Solution we propose consists of two focus areas such as classification and retrieval. A neural network composed of two parallel LSTM layers (to represent query and candidate question), attention layer and a gradient reversal layer (based on domain) is proposed as the question pair classifier. It's trained for individual domains (without gradient reversal) and achieved better accuracy than the latest baseline research for this dataset for 9 out of 12 domains. For retrieval the approach was to retrieve 20 candidates using BM25 and re-rank using classifiers trained already. This selects the duplicate into top 10 with better MAP than BM25 does 6 out of 12 domains. Another important observation is that the common model built with all the data combined gained better MAP than the individual models for 7 domains out of 12 in the retrieval case.

Keywords: Multi domain data, Siamese neural networks, Domain adaptation, Question pair classification, Duplicate question retrieval

ACKNOWLEDGEMENTS

First, I'm grateful to Dr. Surangika Ranathunga for giving me the opportunity and further guidance in selecting and conducting this research. Her continuous supervision greatly helped me in keeping the correct phase in research work. I especially appreciate the frequent feedback on the report, which helped me to correct and fine-tune it to this level. Last but not least, my heartfelt gratitude goes to my parents, wife and friends who supported me throughout this effort.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

NLP Natural Language Processing

QA Question Answering

CQA Community Question Answering

MAP Mean Average Precision

POS Part of Speech

IR Information Retrieval

LTR Learning to Rank

TF Term Frequency

IDF Inverse Document Frequency

CNN Convolutional Neural Network

CHAPTER 1 **INTRODUCTION**

1.1. Background

Searching the web for information has become a day–to-day activity of the general public. Search engines take an Information Retrieval (IR) approach in this, which returns a set of relevant documents ranked by their relevance to the search query. Question Answering (QA) systems take a step further by returning a specific answer by extracting the required information. Community Question Answering (CQA) is a paradigm different to both those. A CQA system is not a search engine that returns a list of ranked relevant documents from which we have to extract the answers, and also it does not try to compose an answer according to user needs like QA systems. Instead Community Question Answering systems directly connect users with the information, provided that the users are willing to share the information. CQA has become popular due to that difference. Both questions and answers are stored for the benefit of future users who search the system for information [1].

The user who asks the question initially searches for similar questions in the system. The user then posts a question only if the search does not return any relevant questions. As shown by Bian et al. [1], the lifecycle of a question in a CQA system has a few stages. First, the user selects a category and then enters a *subject* (title) and a *detail* (description) for the question. After it is posted, the question appears in the system under the specified category for other users to answer/comment on or give feedback. Depending on the site, it is possible to vote for/ give stars for questions, mark duplicate questions, etc. Over the lifetime of a question, if the user finds a satisfactory response, (s)he may mark it as the best answer and may close the question. New answers are no longer accepted for a closed question. In many cases, askers do not close the question and the system itself does it automatically after a period of time, after which the highest number of votes for an answer at the time of a question being closed, determines the best answer.

In CQA systems, novice users who are not able to find the required information properly may ask questions that have already been asked and answered in the system. Some systems give users the opportunity to mark duplicates that they identify. It's common to see questions being marked as duplicates in StackOverflow[1] where they leave a message and are redirected to the duplicate question that is properly answered. However, with the growth of CQA data and the community, users tend to post questions without putting much effort on finding or marking duplicates. This has opened up a requirement of automating this process with efficient and accurate techniques. In CQA systems, pairs of questions become duplicates due to two main reasons [3]. The first and the most common one is paraphrasing, where both users mean the same thing with different sets of words/phrases/sentences being used. The second is different questions that are asked expecting the same answer/opinion, which is harder to identify. Question type can also be a reason for a pair of questions to be syntactically close but to be different in meaning. A main such division of questions is Factoid and Non-Factoid, where factoid questions expect direct answers (facts) and non-factoid questions expect opinions or methodologies [1], which is the type of most questions in CQA.

The lifecycle of a question in a CQA system starts with the selection of a category which is a domain for most large forums. A question raised can be answered by a previous question/answer that has been posted with a different intention and tagged as a different domain. Forums like Yahoo! Answers are not bound to one domain but the problem with that data is that they do not have duplicates labeled which requires a manual effort to use it for validation. StackExchange is a forum where we can find an explicit set of domains i.e. English, Android, Physics, gaming etc. Focus of this research is duplicate detection in a question-answer data set that spreads across multiple domains.

1.2. Motivation

CQA systems have both advantages and disadvantages. Anyone has the freedom to ask questions and expect honest and correct answers. The same freedom applies to users who answer and the responses are seldom moderated. Therefore, it is hard to decide on a correct answer because generally there are multiple answers for a question. As these portals grow in size fast, there is more chance of duplicate questions existing in the database that are not properly captured in searches. Due to these reasons, answer ranking and duplicate detection has become a crucial activity.

According to Doris et al. [5], automatic detection of duplicate questions in CQA data is beneficial because question askers receive answers immediately and the community is not required to flag duplicates any more. There is much research conducted on Community Question Answering and also several were into the area of duplicate detection [3][6][8]. Though they were successful individually, there are fewer attempts being taken to solve the problem for multi-domain data. Some research carried out training and evaluation for individual domains [2][4] that benefitted by domain specific feature learning for better duplicate identification, but it is not the real world setting for a domain neutral data set. And some of the research has artificially generated the required data [6], which cannot be considered as an actual representation of real data based on the distribution of duplicates and domains.

The multi domain data set CQADupStack [5] was built aiming to support multi domain duplicate detection and has made a better foundation to validate our solutions against a common base. Research in the area of domain adaptation for multi-source data is gaining much attention as it is an important problem to solve, especially in the area of image processing and computer vision. But it can be observed that utilization of such novel methods in the area of natural language processing still has much room for improvement. And also the problem of duplicate detection for CQA data is still among active research because it is not trivial with large corpus where percentage of duplicates are low and also with incomplete labeling. Best classification result achieved for the selected CQADupStack data is an average accuracy of 0.92 for one

domain [28]. And the retrieval is still in the average Mean Average Precision (MAP) of 0.17 per domain [5].

1.3. Objectives

The objective of this research is to solve the problem of duplicate detection in community question answering where the data comes from multiple CQA domains. In order to achieve this there are two main sub objectives to be accomplished such that,

1. Identify a better text representation for CQADupStack data that is
    - Domain invariant
    - Discriminative enough
2. Implement a system to do classification and retrieval in multi-domain CQA data more accurately than the existing systems.

1.4. Methodology

Our methodology in brief is to try out and evaluate the techniques that are currently used in similar and parallel research disciplines (i.e. text classification, domain adaptation) to better utilize our objective. Both Classification and Retrieval will be considered for improvement. Classification is to identify whether a pair of questions are duplicates, using latest advancements of text classification. Specially focus on the sequence to sequence binary classification techniques and the dataset is also re-constructed as question pairs. Retrieval is to extract a candidate set of questions (i.e. 10) from the data set and evaluate the rank of exact duplicate (mean average precision). This will be improved for recall and precision by using a technique recommended in the industry (i.e. BM25) to do the initial retrieval and later rerank the candidate set with an improved classifier.

1.5. Contribution

This research introduces a deep neural network-based approach to address the problem of duplicate detection in multi domain community question answering data. It was able to yield state-of-the-art results for 9 domains out of 12 CQADupStack data. CQADupStack is a reference data set for multi domain CQA research which has duplicates labeled.

1.6. Thesis Structure

The structure of the thesis ahead will consist of three main chapters: Literature Review, Methodology, Evaluation and Conclusion. The Literature Review will elaborate much on each fundamental component of our research such as Duplicate Detection and Domain Adaptation, Information Retrieval. Also, it will detail about the specific Data Set that this research is focused on and also about Neural Networks which are identified as the basic methodology that has much room to improve for our data set. The Methodology chapter will be about the approach suggested in this research to address this problem. It will further elaborate on the architecture of the complete system and also the evolution of it throughout the implementation phase. Evaluation chapter will showcase the experiment results and will discuss the possible causes for those results. Finally, it will conclude with our findings and the areas to focus in the future in the Conclusion chapter.

CHAPTER 2 **LITERATURE REVIEW**

2.1. Overview

Research on automating the process of duplicate detection started around the year 2000 [5] when Community Question Answering (CQA) did not have that much popularity. Since then, there were many attempts taken in solving problems of a similar nature that came from different contexts i.e. duplicates in defect reporting systems/bug trackers, question search, etc. Several researches have been done on question search in CQA archives where we can consider it as finding duplicates against a search query [7][9]. All those were not only from academia but from the industry as well.

Duplicate detection is hard due to little word overlap between semantically similar questions. There can be situations where we find a question in one domain being answered perfectly in another domain. Below is an example of the scenarios mentioned above.

New question: (Domain-Health)

Title: Why do bread companies add sugar to bread?

Duplicate question: (Domain-Cookery)

Title: What is the purpose of sugar in baking plain bread?

In this example, we can see the number of words common to both sentences is less and the questions have been asked in different domains. However, both are expecting the same answer, which is the reason why sugar is added to bread.

We can categorize the literature in this research from several viewpoints. Text similarity studies have grown from lexical to syntax and now its focus is more on semantics. Some research takes this as a classification task between relevant and non-relevant entries for a given pair of text. Research which focuses on addressing industry requirements usually takes it as an information retrieval task and tries to predict the best relevance ranking. And further to above there is research which

focuses on natural language tasks for the multi-domain type of data. Following topics in this section will discuss literature from those different aspects and viewpoints.

2.2. Duplicate detection

Techniques for duplicate detection can be categorized into the following major groups considering those approaches.

- Translation and Probability model based approaches
- Syntactic Tree matching approaches
- Vector Space model based approaches
- Neural Learning based approaches

In translation based methods, they use the training to derive probabilities of translation between terms, while in vector space models they find a similarity threshold to determine two question pairs as duplicates. Syntactic trees are a set of rules that construct tree representations for sentences that are later compared for similarity. Neural learning is a technique which enables learning of complex vector representations for text. Identification of duplicates depends on how accurately the vector represents the text which has become the objective of most research in present.

2.2.1. Translation and Probability based approaches

Out of several attempts taken in earlier stages to solve the problem of duplicate detection, work by Jiwoon et al. [6] can be taken as a good inception. It was the time when question-answer systems started to get attention. Before a user query receives a personal response from the forum, this mechanism tries to identify whether a similar question that was asked before exists in the archive. The traditional and naïve methods concentrated more on word overlap. Authors discourage that with some good examples [6]. “Is downloading movies illegal?” and “Can I share a copy of a DVD online” expect almost the same answer but they are lexically very different. Here, a similarity measure that depends on lexical information doesn’t work. There were mainly three different approaches to address the problem at that time. Using machine-readable dictionaries and using manual rules or templates are two of those methods, which are not reliable enough and are expensive. The approach the author has taken was using statistical techniques developed in IR and NLP.

Jiwoon et al. [6] did an important finding about the effect of each segment in a CQA record to detect duplicates. Jiwoon et al. [6] has shown that retrieval of relevant questions by considering only the title gives better mean average precision (MAP) values irrespective of the retrieval model used (here it’s query likelihood language model LM and Okapi). Jiwoon et al. [6] generated their training samples by using the similarity of answers with a statistical method. However, a training set in which duplicates are determined by human judgment would be more appropriate in this kind of experiment.

Huizhong et al. [7] took this problem as a Question Search problem. Their intention was to return a set of questions that are semantically equivalent or close to a queried question. They proposed a method that goes beyond the syntactic equivalence of text and identifies the question topic and question focus to search for similar questions. By applying this methodology to a large collection of data taken from Yahoo! Answers, they were able to show that it outperforms the Vector Space Model and Language Model for IR, which were taken as the baselines. This research is specific due to the novel approach they took in identifying the question focus and question

topic. It has built a hierarchy of semantic importance of each part in text. This research is more focused on finding questions relevant to a given query that slightly deviates from the task of duplicate detection among a set of posted questions.

One of the latest attempts in translation based question retrieval in CQA data is the one by Amith Singh [9], which is special due to the introduction of an entity catalog that does the translation. He identified that the earlier approaches [6] used translation models only to give different representations for the text of the questions. According to him, they highly depend on the quality of the corpus used and loses context information. The proposed approach, which is known as Entity based Translation Language Model (ETLM), builds semantic relationships between entities and words. They have chosen Wikipedia as the entity catalog due to its huge coverage and timely updates. Using the entity catalog they annotated both the QA corpus (offline) and the incoming query (online). They used a data set crawled from Yahoo! Answers. They took a supervised learning approach where they got the data annotated by human annotators for relevancy against a set of queries. The training phase of this method is to learn relationships between entities and words of the used corpus. It derives translation probabilities of a given term into another that ultimately sums up and gives a score for relevancy of a document to a given query.

2.2.2. Syntactic Tree matching approach

Syntactic tree represents a textual phrase according to grammar, which defines possible word arrangements in a language. This property has been used by Kai et al. [11] from which they achieved an improvement of 8.3% MAP than when using lexical similarities. This technique is different from what we discussed earlier because it classifies the data according to a predetermined set of structures (syntactic tree). Weights of similarity between two tree fragments depend on two factors; the depths of the fragments in the tree and the sizes of the tree fragments. If a large portion of trees is the same, then the weighting factor based on fragment size becomes high because when the fragment is large, then the varied information it carries is also high. The bottom level tree fragments are also carrying more semantic

information than the higher levels. Node matching score between two trees is the multiplication of the weights of all the matching fragments under those nodes. Higher node matching scores indicate higher similarity between those node pairs. With Yahoo! Answers data, they have shown that 5~12% improvement of MAP for similar question identification can be achieved with this method. There's no specific training phase involved, and this method is based upon rules represented by syntactic trees.

2.2.3. Vector Space based approaches

In Multi-Dimensional Vector Space, modeling a document (question) is representing it as a multi-dimensional vector, where each dimension corresponds to a word in the data set. The weight of a dimension was determined by the frequency of the word in the corresponding text. Term frequencies are dampened as equation (1) to keep a normalized impact on the vector representation of text.

$$weight = 1 + log(frequency) \quad (1)$$

An improved weight measurement is used by Wang et al. [2]. In their research, as shown in equation (2,) inverse document frequency is also considered in calculating the weight.

$$weight = tf_i \times idf_i \quad (2)$$

$$idf_i = log\ (Dsum / Dw_i) \quad (3)$$

In formula (3) Dsum is the total number of documents and Dw_i is the number of documents that contains the i-th index term.

Cosine similarity is a measure used in models that represent text as vectors. Similarity value can be derived by formula (4) where W_{1i} is the i^{th} dimension of the 1^{st} vector.

$$Sim = \frac{\sum_{i=1}^{n} w_{1i} w_{2i}}{\sqrt{\sum_{i=1}^{n} w_{1i}^{2} \times \sum_{i=1}^{n} w_{2i}^{2}}}$$

(4)

Runeson et al. [4] researched a mechanism to identify duplicate defect reports, which was very good research from the industry. There, Runeson et al. used a vector space to model the data. Along with a vector-space model, they used cosine similarity as the measure of similarity. They also took data preprocessing steps such as tokenization, stemming, stop words removal, synonyms and spell checking. With all these variants, they concluded that about 40% of the marked duplicates could be found with the methodology they followed. This research was a good reference to identify how fundamental IR techniques are applied in CQA duplicate detection. Runeson et al. [4] confirmed the observation that Jiwoon et al. [6] did about the difference in importance of each part of the question. Runeson et al. [4] showed that giving twice the importance to the title than the description gives a significant improvement in recall. Marc et al. [8] in their experiment for duplicate detection in Qatar Living forum data, achieved better results compared to the others, where they measured similarities in body, title and full question level. The data set was limited to defect reports of the products they built where they had the advantage of domain knowledge.

Conventional bag of words based similarity measurements considered word overlap, which evaluates the lexical features of the text. It's not encouraged to use those lexical techniques due to the lack of semantic representation they have. However, it's evident that their usage has also contributed to better accuracy in research such as the winning team of SemEval 2016 – Community Question Answering task [8]. They have considered a combination of lexical and semantic features where word overlap, noun overlap and n-gram overlap were measured.

2.2.4. Neural Learning based approaches

As we discussed in 2.2.3 of this chapter it's evident that vector representations are depending on lexical properties of text (i.e. word frequency, n-grams). But in order to evaluate the similarity between two texts it needs a much better vector which is able

to represent the semantics also. There are parameters which cause the correctness and the bias/weight applied on appropriate parameters make significant improvement in the final representation. In order to fit the models to validation data those parameters should be tuned optimally. And also the parameter values that are optimized for validation data may be overfitting and might give poor results on test data sets. This is where Neural Learning comes into picture. In modern ML approaches the parameters are tuned automatically and avoid overfitting as well.

Distributed representations of text

This is an improved version of vector representation of text introduced by Mikolov et al. [10], which is used by most of the recent text similarity related research such as duplicate detection. Earlier vector representation models have lost information of connection (i.e. similarity/difference) between words, where they treated words as simple atomic units. That also has their own advantages like simplicity and robustness in training and using those models. However, with the progress of machine learning techniques in recent years (i.e. neural networks) more complex models have become feasible to train efficiently. As they say, there are multiple degrees of similarities between words that go beyond the syntactic regularities. Authors have explained with an example such that vector(”King”) - vector(”Man”) + vector(”Woman”) results in a vector that is closest to the vector representation of the word Queen. In their research, they build model architectures that preserve such regularities in words and show those can be learned with high accuracy.

A most recent attempt taken to solve a problem of a similar nature was in SemEval-2016 workshop[1]. Subtask B of task 3 in the above workshop was on Question-Question Similarity where researchers were interested in the ranking of relevant questions from a data set against a given question. The data was taken from the Qatar Living forum[2].

[1] http://alt.qcri.org/semeval2016/task3/

[2] www.qatarliving.com/forum

The team that recorded best accuracy, UH-PRHLT [8], took both lexical and semantic-based measures to represent data. They used word n-grams and bag of words as lexical representations. For semantic representation, they employed four main techniques. First, is distributed representations of text, which is discussed earlier. Next is a word alignment strategy, which considers the alignment of the words of two texts to determine the relatedness between those. Knowledge graphs and common frames are the other two semantic techniques.

2.2.5. Pre-trained Models

Usage of pre-trained language models which support multiple NLP tasks that are trained with large collections of textual data was also an option for duplicate detection. Bidirectional Encoder Representations fromTransformers (BERT) is one such implementation by J Devlin et al. [38] from Google AI team which got huge attention among the community recently. It was able to give breakthrough state of the art results for eleven NLP tasks. For a pre-trained model to be useful it should be finetuned with a novel data set for different tasks. Authors of BERT argue that the existing pre-trained models cannot fully utilize their strength when it comes to fine tuning. According to them the problem with standard language models is the unidirectional nature which has overcome in BERT by using a masked language model (MLM). The basic idea of MLM in BERT is masking a percentage of words in sentence and predict them in training. This will use the information from the entire sentence both left and right context to predict the masked word. Different of this approach against the bidirectional LSTM based language models is that they are using the information from subsequent words to predict the masked in both left to right and right to left directions.

BERT approach was a game changer but it had its own flaws which were not there in traditional language models. The approach which BERT used is autoencoding based pretraining and the traditional approach is autoregressive (AR) language modeling. AR modeling is a feedforward model which predicts a future word based on a set of words given in the context and that context words constrained to either forward and backward directions. Z Yang et al.[39] addressed these problems very recently with

their new algorithm XLNet which has outperformed BERT in 20 NLP tasks. According to the authors XLNet is best in both worlds because it retains the bidirectional context capturing method which BERT introduced while avoiding the training algorithm from predicting the mask with parallel independent context. Instead they propose permutation language modeling where it predicts a token based on preceding context like in traditional approach but the order of token prediction is random and not necessary to be sequential. XLNet has become the cutting edge pre-trained language model by giving state of the art results for most NLP tasks with large margins from the predecessors.

2.3. Information Retrieval (IR)

The most abstract and industry-oriented research discipline that we can categorize this problem into is IR. CQA duplicate detection is retrieving similar questions from a data set of question answer pairs. But it has its own characteristics which differs it from another information/text retrieval application. Ad-hoc retrieval is such an application which searches for documents that are relevant to a user who specified the information needed by a query and returns a ranked list according to the relevance. But in ad-hoc retrieval, data is a set of documents, but in CQA it is short text question answer pairs. The field of study “automated question answering” is also another application that has an influence on this but it is more in to answering factoid questions. CQA is better than QA systems in answering non-factoid questions like opinions. However, there’s more delay incurred in getting a question answered in CQA systems than its counterpart for which we have to rely on the community.

Similar to most of the IR problems, this is also a ranking problem by nature. As we cannot expect the model to return only the exact duplicate all the time, retrieved potential duplicates should be ranked according to the relevance. Here in this section we discuss differences of ranking applications and ranking models. But when we go into detail, the categorizations are more focused on the way they calculated the text

similarity which eventually determines the rank of a document relevant to a given query. We have discussed methods which calculate text similarity in chapter 2.2

2.3.1. Ranking applications

Community question answering is one among several text retrieval and ranking applications [15]. Applications are varied mostly based on the data sets and the nature of queries. Here we brief a few such applications which have the closest relation to CQA and help to understand a model for CQA.

Ad-hoc retrieval is the most general scenario where a user specifies the information needed as a query with few words or a few short sentences which includes keywords. The set of searchable documents ranges vastly on different aspects. Length is from short sentences to several chapters, different authors, different vocabularies, different semantics are few of such variants. The term ad-hoc is used because the document set stays relatively static and new queries are submitted. One major characteristic of ad-hoc retrieval is the heterogeneity of the queries and the document collection.

Question Answering applications tries to find out the answers to questions asked in natural language by users where ad-hoc retrieval is giving only a set of relevant documents to extract the expected information from. In QA the heterogeneity of information and queries is much lesser but the queries are mostly natural language sentences than a set of keywords.

Next is Community Question Answering systems on which this research is focused. As we already discussed in the Introduction, CQA attempts to find a similar question from QA pool where it has much better homogeneity between input query and target which is also a short query sentence. Therefore, an improvement for the existing techniques should pay much attention to semantic equivalence and vocabulary mismatch than searching for keywords in QA pool.

2.3.2. Ranking models

According to Tie-Yan Liu [16] we can categorise the ranking models as query-dependant and query-independant based on how they relate the query for ranking.

Here we showcase two prominent examples for query-dependent and query-independent models for information retrieval.

Query-dependent models

Models in this category ranks documents based on their relevance to the input query. Approaches in the category differentiates each other by the way they calculate the degree of relevance. Vector space and probabilistic models are two major variants of such methods of calculating the relevance. Okapi BM25 [33] was one of such methods which was a state of the art for information retrieval in the decade of the 90's. BM abbreviation for Best Matching and Okapi is the first IR system used as the retrieval algorithm. BM25 uses a TF-IDF like retrieval function which developed based on probabilistic retrieval. Most general BM25 equation for a relevant score of a document is as follows.

$$\mathrm{BM25}(d,q) = \sum_{i=1}^{M} \frac{\mathrm{IDF}(t_i) \cdot \mathrm{TF}(t_i,d) \cdot (k_1+1)}{\mathrm{TF}(t_i,d) + k_1 \cdot \left(1 - b + b \cdot \frac{\mathrm{LEN}(d)}{\mathrm{avdl}}\right)},$$

In the above equation it calculates over each query term from *i* to *M.* k1 and b are free parameters to be chosen accordingly to optimize the algorithm. LEN is the length of the selected document and avdl is the average length of a document in the set. TF is the term frequency of a query term in the selected document and inverse document frequency is calculated for a term with respective to the document set. These parameters are described individually in Chapter 2.2.

Query-independent models

Here in this approach documents are ranked based on their own importance. PageRank [17] is one good example for this which is mainly used for web search. PageRank was the main algorithm used by Google initially to order the search results. It uses the hyperlink structure of the web to determine the importance of a web page. Assumption behind this algorithm is that the importance of a web site/page depends on the number and quality of the links that are received from other

web sites/pages to it. PageRank value for a certain web page is derived by below basic equation which also has other variations that handles practical scenarios.

$$\mathrm{PR}(d_u) = \sum_{d_v \in B_u} \frac{\mathrm{PR}(d_v)}{U(d_v)}.$$

It's calculated by summing up the values derived by dividing PageRanks of each page that are having links to considered page by the number of outlinks from each of such linked page.

2.3.3. Neural Ranking

According to Tie-Yan Liu [16] Learning to Rank for IR is a task to automatically construct a ranking model using training data, such that the model can sort new objects according to their degrees of relevance,preference, or importance. J Guo et al. [15] has defined the learning to rank problem in a more generalized formulation with respective to neural learning as below. Its two parts are the loss function and the ranking function.

Loss function

$$f^* = \arg\min \sum_i \sum_j L(f; s_i, t_{i,j}, y_{i,j})$$

Here f is the function that evaluates the relevance score for a query document pair. s_i is the query and $t_{i,j}$ is the set of documents that are associated with the query. $y_{i,j}$ is set of actual relevance score / ranking labels associated with the query. Therefore LTR is finding f^* the best version of ranking function which minimizes the loss function L.

Ranking function

$$f(s,t) = g(\psi(s), \phi(t), \eta(s,t))$$

Ranking function evaluates the rank of a document t against a query s. ψ and φ are functions which extract features from query and document. η is the interaction function which extracts features from query-document pairs and defines how the matching between query and document is evaluated. g is the function which evaluates the relevance score. Traditional approaches used static functions for feature extraction but with machine learning those functions also get learned from training data.

Due to the recent growth of deep neural networks especially because of the computer vision applications we can see that NLP applications also highly benefited. Neural networks have sufficient capacity to learn the parameters of relevance estimation than most of the methods that are based on hand crafted features. We have already discussed the Neural Learning based duplicate detection approaches in chapter 2.2.4 because it's essential for understanding Neural Ranking for CQA. Guo et al. [15] in his research has come up with a comprehensive study of neural network based information retrieval models from Architectural, Learning and Training point of views. In the following subsection we will brief the Neural Ranking categorizations of that research because it helps us to understand where exactly the CQA duplicate detection applications are positioned in the neural ranking problem space.

Learning Objectives of Neural Ranking

Learning objective is how the loss of ranked result is being calculated. This is an essential part in the learning phase of a neural ranking model because iteratively decreasing the loss will eventually make the optimal version of ranking function. Each technique has its own advantages and disadvantages.

Pointwise Ranking Objective

A set of query-document pairs will be considered for evaluation. Loss will be calculated for each query (s) and document (t) pair independently by comparing the rank derived from the model and taken from human assessors. Here it tries to optimize the ranking labels of each document for query. But this objective is said to be less effective in ranking tasks because it does not consider document preference or order information.

Pairwise Ranking Objective

Here it considers all the permutation of document pairs for their relative preferences. Loss will be calculated for each document pair based on the rank difference it has got for each query. Most preferable documents for a given query will get a higher ranking. This model is also not optimal for a ranking due to two reasons. Because it's impossible to construct a ranking model which can predict preference for all cases and also all the question pairs are not equally important for ranking.

Listwise Ranking Objective

To address the issue of two above approaches facing in ranking task most of the research has proposed Listwise Ranking as a better choice. Here the comparison will be between ranking lists. Derived ranking list of a query will be compared with the candidate document list of the same query. This objective reflects the final ranking performance of the model.

Multi-task Learning Objective

The aim of this approach is for the model to use information from one domain to understand the information of another domain when optimizing the neural ranking. This way ranking functions will be trained to make representations of data that cannot be discriminated against by domain. Domain independent patterns which are useful in cross-domain applications will be learned. Therefore this objective aligns well with the multi domain duplicate detection. This approach will be further discussed in Section 2.4.2.

Training Strategies for NN

These variations are based on the amount of labeled data available. The three categories available are Supervised, Weakly Supervised and Semi Supervised. In Supervised learning there will be sufficient data which is labeled by human assessors to train a model. But most of the time there will only be a limited amount of expert annotated data and researchers are learning how to constrain the parameter spaces that should be learnt. In the Weakly Supervised category, the data is annotated by an existing industry accepted technique like BM25. Semi Supervised approach is taken when there's human annotated data but not a sufficient amount. This is the scenario which most research encounters practically. Fine tuning the weakly supervised models and controlling the learning rate of them using available annotated data is two such applications of semi supervised learning.

2.3.4. CQA Neural Ranking

The aim of most of the research in the field of text retrieval is constructing a better version of text representation. When it comes to Neural ranking, the most successive attempts are taken for creating embeddings for given text. Success depends on how accurate the vector representation being learnt by the proposed model. Trending topics in this domain are mostly the different flavors of neural networks. Similarity evaluations are still conducted with fundamental techniques like cosine, Euclidean, Manhattan distances and inner product of vectors etc. distances. Another practice that is popular among retrieval based research is the usage of a standard retrieval model i.e. BM25 to query a small candidate set of questions and later applying relevance ranking on that which makes the training more efficient [29]. Following sections discuss a few of such research that showed better results in neural learning for information retrieval and ranking.

Convolutional NN

Qiu et al. [22] took a different approach not just because of the application of CNN to model the sentence representation, but because they used a neural tensor layer to determine the question relatedness as well. Another specialty about this research is that they feed both the questions and answers in the training network by which they

expect a better learning of semantics. Their setup for training starts with a lookup layer that encodes all the word tokens into vectors. They used Word2Vec word embeddings by Mikolov et al [10] to initialize the parameters. Then the encoding of questions and answers into fixed length vectors is done by convolutional and pooling layers. Finally a tensor layer decides the interactions between them. Convolutional neural tensor network is a general architecture which has improved around identification of images therefore the adaptation of it for this NLP task has not required complicated nlp pre-processing and prior knowledge.

Recursive NN and LSTM

Recursive neural network is tried for information retrieval tasks by Hamid et al. [23]. They found out that the model evolves over time and gets trained to take only the useful information from any new input. This research was conducted for web document retrieval from which has good insights for CQA data also. According to the authors, it is more important to learn sentence embeddings than word embedding for a task where it computes the similarities between text strings. In their architecture, they accept words of a sentence in a recurrent manner where they map each into the latent space with the historical information. As it reaches the last word of a sentence,n the hidden activation forms an embedding vector to represent actual. Long Short Term Memory (LSTM) cells are introduced to incorporate the learning difficulties of long term memory in RNN. Final relevances are calculated based on cosine similarity. They show that the LSTM-RNN is effective in learning keywords while reducing the effect of less important words.

Intention of the research Zhiguo et al. [24] conducted was also into text similarity but their focus was to improve the calculation of similarity than coming up with a better version of text representation. They mainly considered two things in matching such as Bilateral and Multi Perspective. When comparing two sentences P and Q they took the similarity from both directions P->Q and Q->P and matched it with different perspectives of representation such that character composed embeddings and word embeddings. Further to that they have involved multiple contextual embeddings also in vector representation. In order to match between those multiple perspective

combinations of text they used four main matching strategies as well. i.e. Full, Max pooling, Attentive and Max Attentive. The final text similarity evaluation of this model is based on the probability distribution of all combinations.

Siamese NN

Mueller and Thyagarajan. [25] tried to address the same problem using a Siamese NN focusing on the scarcity of labeled data. Siamese means twin and here in Neural Networks tries to derive a single output based on two input vectors. This is the main reason that causes Siamese to be successful in addressing comparison of two texts for similarity. Mueller and Thyagarajan [25] used synonyms found in Wordnet by Miller [41] to expand their data set by randomly replacing words of existing data which is known as synonym augmentation. They also used pre trained 300-dimensional Word2Vec embeddings by Mikolov et al [10]. The newly proposed method in this research is Manhattan LSTM which uses tied weights of two LSTMs in siamese architecture. They claim that this aggregation is more useful in applications of asymmetric domain for information retrieval.

Neural Attention

This is a mechanism especially used for neural machine translations to predict the translated version of a given text. The attention mechanism proposed by Bahdanau et al. [37] is one of the prominent neural attention mechanisms. In a machine translation mechanism the main components are encoders and decoders. As Bahdanau et al. [37] discuss, in traditional methods of the encoder-decoder model, the segments are hardly coupled for translation and also they were trying to compress the information into fixed length vectors. But here in the approach they came up with, it lets the network to search the most relevant segments that help predicting the target word. It gives attention to the most relevant parts of the input when predicting output sentence. Internally it takes two sentences and turns them into a matrix, where one sentence represents rows and the other represents columns. Then it makes matches, identifying the relevant context. The context vector is the weighted sum of all past states of the encoder. Figure 2.1. from Bahdanau et al. [37] research illustrates how the matrix, which is described earlier, maps the words of two related

sentences of English and French. The gradient of the color is the representation of how important a word of input is to produce a word in output in a translation task.

In the community-based information sources especially in CQA data, there are significant amounts of redundant and irrelevant data that does not help in a task like information retrieval and also adds noise to the ML algorithms as well. Salvatore et al. [26] in their research claims that attention-based pruning of neural learning algorithms gives better results while improving the running time (because it prunes the trees node significantly). They used LSTM networks for question retrieval along with Tree Kernel (TK) based reranking. The aim of the attention model is to feedback the LSTM with the most important pieces of the text. Because of the lesser usage of text authors gained a five times improvement of speed than the models used in entire sentences.

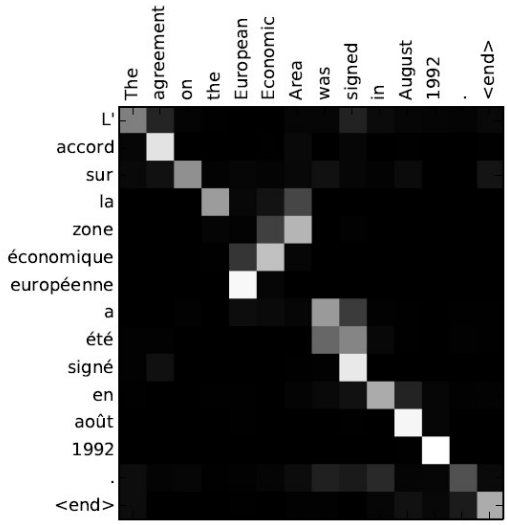

Figure 2.1. Neural attention weights of an English-French translation [37]

2.4. Domain Adaptation

As we discussed in chapter 2.2.4 neural learning is much more effective in capturing new features that are not observed in traditional models that use hand crafted features. But this advantage has its own drawbacks also. Learned features and

models in this way are highly bound to the domain where the training data lies. Neural networks are very data greedy applications. Therefore when there's an absence of data to achieve a certain learning task we should have to depend on a similar distribution that has labeled data. Adversarial Learning is found to address this requirement. It behaves as a cross domain regularizer which will try to predict the domain and avoids the model from learning domain specific representations.

2.4.1. Adversarial Domain Adaptation

The inception of the research on this aspect was built around computer vision and image processing tasks. Later NLP also adapted techniques from those research in order to achieve tasks of different domains. Below are the prominent and fundamental adversarial domain adaptation approaches in the literature. The Domain Adaptation is described here as the usage of a model which is trained with data from one domain to resolve a similar problem in another domain.

Using gradient reversal layer

According to Ganin and Lempitsky [18] Domain Adaptation is learning a discriminative classifier or other predictor when there is a difference between training and test distributions. They mainly focused on combining domain adaptation into the learning process of data representation. Underlying principle of this architecture is to optimize both discriminativeness and domain invariance. This is achieved by minimizing the loss of label predictor in a classification task while maximizing the loss of domain classifier that discriminates between source and target domain. In this approach the fundamental addition to the usual deep neural network setting is gradient reversal layer that is shown in the Figure 2.1 which is taken from their paper [18]. Domain classifiers will add a negative gradient value to the feature extraction phase of neural network in backpropagation. It doesn't require any change in the input instead it multiplies the gradient in back propagation by a negative scalar.

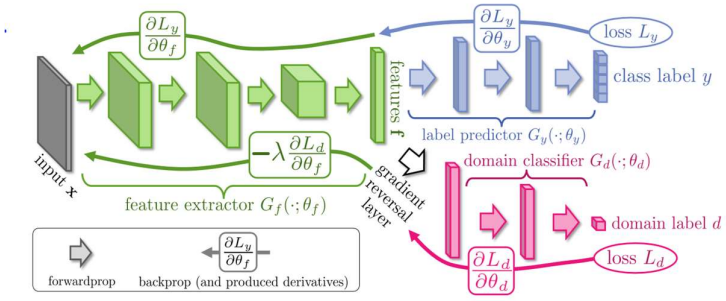

Figure 2.2. Domain adaptation by Gradient Reversal [18]

Using generative adversarial nets (GAN)

This is another popular approach which Goodfellow et al. [19] proposed for domain adaptation. GAN uses two neural networks which are competing with each other simultaneously to generate samples that are domain invariant. Two models are namely a generative model (G) and discriminative model (D). Generative model tries to capture the properties of data distribution while the Discriminative model tries to predict the probability of the sample coming from training data. Competition is that G is trying D to make a mistake (maximizing the domain discrimination loss). Eventually this will end up generating samples that are domain invariant and also effectively represent the distribution.

Using deep domain confusion

Objective of this method is also similar to the two above approaches. Tzeng et al. [20] optimized a deep convolutional neural network (CNN) for both classification loss and domain invariance. Here the domain confusion term is used to ensure that domains are indistinguishable in the learned representation. They have introduced an adaptation layer and a domain confusion loss to the general CNN. They used Maximum Mean Discrepancy MMD[1] as the domain confusion metric. MMD is a kernel based approach aimed at measuring the distance between two probability distributions.

Generalized approach uses GAN and domain discriminator (ADDA)

Tzeng et al. [21] made a recent breakthrough for three above approaches which were recognized for some time in the field of adversarial domain adaptation. They argue

that though the above work is valid but are not general enough to tackle the problem. As they say GAN based approaches are not optimal for discriminative tasks and can be limited to smaller domain shifts while discriminative approaches impose tied weights on the model and do not exploit GAN based loss. Adversarial Discriminative Domain Adaptation (ADDA) [21] is the proposed architecture to address mentioned issues. This approach can be considered as a combination of prior research and they claim that it is generalized, simple and effective. Existing approaches can be considered as components of this because it applies each such method based on the input data. It learns a discriminative mapping of target data to the feature space of source data which is taken as the deciding factor. This approach is categorized as unsupervised because it is optimized for the scenarios where there are no labeled data in the target domain.

2.4.2. Adversarial Learning on Multiple Domains

Adversarial learning is gaining huge attention and research a lot but most of those focused on improving learning algorithms for single-source single-target setting. Zhao et al. [30] tried to address this multiple source domain adaptation in their research. To achieve it they proposed a multisource domain adversarial network (MDAN). This research is a theoretical analysis and also conducted on classification and regression tasks. This is important because there are very few with this problem combination. The fundamental technique they used is backpropagating the domain classification error into neural networks with gradient reversal. In order to learn for multiple domains, they have come up with two flavors of this where namely Hard and Soft. In Hard version the source that has got minimum domain classification error will be back propagated while in Soft version all the classification errors are aggregated to backpropagate. Authors claim that MDAN outperforms state-of-the-art domain adaptation methods for three mentioned real-world datasets in classification and regression tasks.

Cohen et al. [31] took a similar approach for ranking tasks where they used an adversarial discriminator to predict the domain. It also followed gradient reversal mechanisms to feed negative feedback avoiding models from learning domain

specific representations. Here the discriminator acts as a domain predictor. The loss function (domain identification loss) is supposed to maximize when the model learns better domain invariant models.

Multi-Adversarial Domain Adaptation (MADA) is a novel technique which is proposed by Pei et al. [32] that focuses on using multiple domain-discriminators. This approach differentiates itself from the other counterparts by its ability to capture complex multimode structures which exploits fine grained alignments of multi domain data distributions. This architecture uses k number of discriminators when there are k number of domains. The effect of each discriminator on evaluated data is distributed over the domains based on the probability distribution of class labels. They argue that MADA can avoid negative and under transfers which were disadvantages of earlier transfer approaches. Negative transfer occurs when the modes of distributions among domains are false aligned and under transfer is caused by when the matching of such distributions cannot be maximized. Therefore they say that MADA is the way that can promote positive transfer and reduce negative/under transfer.

As we discussed earlier in this section multi adversarial models focus on learning commonalities of multiple data sets. But Liu et al. [35] in their research shows that the learned common models in most of such approaches are contaminated by the features or noise of other task/data-sets. They propose Adversarial Multi-task Learning in which it's expected to learn a much better model that gives higher accuracy and efficiency for sub domains than the models learned individually. It improves the accuracy of subdomain tasks with the help of data that comes from other domains. They have shown two flavors of this architecture based on the shared lstm layer as depicted in Figure 2.2. borrowed from their research paper [35]. In the Fully Shared model the same lstm layer is shared by all the sub tasks. In the Shared Private model there are LSTM layers to represent both the shared and private spaces. The speciality is the orthogonality constraint which is applied to discourage redundant latent representations (i.e. noise) and make the shared and private layers to model different aspects of the input.

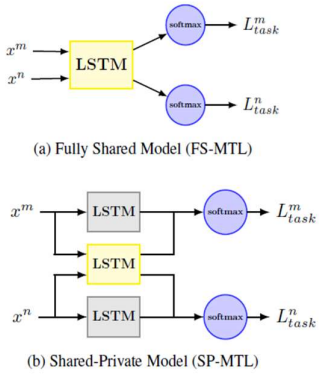

Figure 2.3. Multi-task learning architectures [35]

Multinomial Adversarial Networks (MAN) by Chen et al. [36] is also an attempt taken to address the problem of generating a model that works for multiple domains by reducing the divergence of feature distributions among each of those. As this was conducted on a text classification task the theoretical aim was to simultaneously minimizing the differences among multiple probability distributions. They also used a text classifier and a domain discriminator alongside. The difference was in the data feeding mechanism where they used mini batches of data from each domain to go through the setup iteratively.

2.4.3. Adversarial Learning in Retrieval

Earlier in this chapter it's discussed research conducted on adversarial domain adaptation and also about neural learning but it's interesting to know how they applied for domain adaptation in retrieval to address a real-life scenario. Because the approaches which are proven in classification and regression setups should finally be adopted into question retrieval applications.

Jianfei Yu. et al. [27] has worked on a much practical application of which is similar to what we discuss here. It's the chatbot system of Alibaba's Ecommerce customer service. Though they have named it as a QA system it has the nature of CQA which uses a knowledge base to obtain the nearest question for a question raised by a user. In practice the domain of incoming questions are not defined or limited. Therefore it requires a method of transfer learning for domain adaptation. An important extract from their research is the way they used prior research findings in actual application.

They are indexing all of the questions about their knowledge base with Apache Lucene[1] . When a query question comes in it searches for top-k similar questions using a TF-IDF ranking algorithm. Then the Reranking of that candidate question set was addressed as a Paraphrase Identification task which has made it an efficient retrieval model. The sentence representations that are derived by their model will be fed to an existing gradient boosted regression tree algorithm for reranking. As the modeling framework they proposed a hybrid approach which aggregates a sentence encoding cnn and a sentence interaction cnn that complement each other to derive the final model. For the transfer learning part they proposed the usage of covariance matrix that models the domain relationships. They argue that the CNN based methods are much more time efficient than existing LSTM based methods and also they claim that the transfer learning model they use gives better results compared to other methods for source-target pairs they used.

2.6. Data sets

Yahoo! Answers is the most popular CQA archive among researchers due to its diversity and the amount of data it contains. Huizhong et al. [7] used two of the top-level categories at Yahoo! Answers, namely, 'travel' and 'computers & internet'. They took manual judgments to assess the relevancy of returned results to a queried question. A. Singh [9] also took a similar approach with Yahoo! Answers with manual relevance judgments. Both these experiments were question retrieval/search type, which is slightly different from a duplicate detection within a given data set. Jiwoon et al. [6] used data from Naver[3], which is a leading portal site in South Korea. In order to train their translation based retrieval model, they collected similar questions by comparing the answers of each question. They assume that if two answers are similar then the questions also should be similar. This way they grouped their question according to their similarity, which didn't involve a human judgment. Wang et al. [2] used the bug repository of Eclipse[4], a popular development

[3] http://www.naver.com

[4] https://eclipse.org/

environment, to calibrate their approach and used the bug repository of Firefox[5], a popular web browser, to evaluate it. Runeson et al. [4] also got a data set of a similar kind from Sony Ericsson Mobile Communications. That data is from their defect management system. Both above researches took their data from a technology domain, which gives duplicate detection a leverage by the nature of data.

CQADupStack

StackExchange[6] is a network of question and answer websites on topics in varied fields (i.e. gaming, mathematics, programming, and physics). Stack Overflow[7] is one of those fields/sites that is widely used by programmers.

CQADupStack data set [5] is a multi-domain data set from StackExchange that has the duplicates labeled. It also has helpful utility functions for data manipulation. The version of the StackExchange, which is dumped to create this data set, consists of 149 sub forums. The authors have made a few criteria, which filtered most of those sub forums that doesn't help a duplicate detection research. Forums with at least 10000 threads and 500 duplicate questions are considered to be worthwhile to have in a data set because if the numbers are lesser than that there's no special advantage of automating. The sub forum StackOverflow was also discarded due to its size, which is significantly larger than the average sub forums. Finally, 12 sub forums are selected to build the duplicate detection data set. Table 3.1 below gives a brief but good overview of the content of the selected sub forums.

[5] https://www.mozilla.org/en-US/firefox/new/
[6] http://stackexchange.com/
[7] http://stackoverflow.com/

Subforum	# threads	Ave. # answers per question	Ave. # words per question	Ave. # words per thread	% duplicates	Ave. # dups per dup question
android	23697	1.73	100.4	211.2	7.23	1.08
english	41791	2.74	83.4	338.0	9.31	1.11
gaming	46896	1.85	85.5	276.8	4.86	1.03
gis	38522	1.67	115.8	227.7	2.90	1.02
mathematica	17509	1.88	120.0	266.4	7.84	1.08
physics	39355	1.91	154.5	517.5	5.00	1.11
programmers	33052	3.89	166.5	740.1	5.26	1.13
stats	42921	1.65	160.1	356.0	2.13	1.03
tex	71090	1.61	95.3	199.2	7.31	1.05
unix	48454	1.89	102.6	249.2	3.54	1.04
webmasters	17911	1.87	109.4	287.2	7.79	1.22
wordpress	49146	1.52	104.7	194.9	1.52	1.04

Table 2.1 Summary of the CQADupStack data set [5]

Generally, the percentage of questions marked as resolved is low. This makes an automatically found duplicate question to be of no value to the end user. Authors of CQADupStack have suggested considering the number of up votes as a parameter to decide on the correct answer.

The authors of the CQADupStack data set have preprocessed the data and have kept only the information relevant to duplicate detection. The data format has been changed to a lightweight JSON representation, which was in XML. Researchers are also equipped with the Python 2.7 script to do data manipulations. The script helps in tasks such as the removal of links/stop words/punctuation, stemming (NLTKs' Porter stemmer), expansion of contracted forms etc.

For the experiment purpose we should construct data sets out of the raw data given. For classification purposes we need to have a data set of question pairs where the labels are either Duplicate or Non-Duplicate. As the authors propose we need to pair each question with all the questions chronologically precedes that. This way the question pair data set will preserve the real-world nature of it but become greatly imbalanced because the number of duplicate pairs is much lower compared to the number of non-duplicate pairs. Therefore they have suggested constructing the classification set with more balance (1:10 Duplicate: Non Duplicate) to give much importance to positive data. Liang et al. [28] who has achieved the best classification results for CQADupStack so far has used a question pair set that is having Duplicate vs Non Duplicate ratio of 1 : 3. In the case of retrieval also we have to consider the

imbalance nature of data. There can be a lot of queries that return empty lists for duplicate retrieval. Current result sets don't handle the queries for which the result set is empty [5]. Therefore, authors of CQADupStack have reported baseline results for retrieval scenarios considering only the queries that are having duplicates for.

2.6. Evaluation & Baseline

In this section the discussion is about the best results achieved so far and also how they are being measured. This will help us to decide on a reference, against which we can compare our results. Therefore the focus of this section is to identify how we validate this research.

2.6.1. Evaluation

In this section we discuss the evaluation measures usually used in IR and text similarity activities. The definitions are from the book An Introduction to Information Retrieval [13] by Manning et al. and research by Tie-Yan Liu [16]

Precision – is the fraction of retrieved documents that are relevant

Recall – is the fraction of relevant documents that are retrieved

F1-measure – is the weighted harmonic mean of precision and recall

Accuracy – is the fraction of its classifications that are correct

But to evaluate a ranking task the position of a result in a list of results should also be considered. In order to evaluate a ranking result few fundamental steps should be followed. Initially there should be a randomly sampled collection of queries against the data set being considered. Next is to get the relevant judgments from human assessors. Then rank the documents with a model and measure the difference against the human evaluation using an evaluation measure. These values should be averaged over all the queries in the test set in order to get the performance of the ranking model. Below are such measures which were constructed specially for IR Ranking tasks.

Mean Average Precision (MAP)

This is the primary and most commonly used one among these measures. Average Precision is the average of the precision value obtained for the set of top k documents existing after each relevant document is retrieved, and this value is then averaged over information needed to get MAP.

Precision of a query should be defined at a position k as below.

$$P@k(q) = \frac{\#\{\text{relevant documents in the top } k \text{ positions}\}}{k}.$$

Then calculate and sum up all such precisions over possible k values which goes from 1 to the total number of documents associated with the query. That will give an average precision value by the below equation. Here l_k is the binary value for relevance of the document at k^{th} position.

$$\mathrm{AP}(q) = \frac{\sum_{k=1}^{m} P@k(q) \cdot l_k}{\#\{\text{relevant documents}\}},$$

Taking the mean of AP values of all the test queries will give mean average precision.

Mean Reciprocal Rank (MRR)

It is defined by Burges et al. [14] as that if r_i is the rank of the highest ranking relevant document for the i^{th} query, then the MRR is just the reciprocal of that rank, averaged over queries. In the equation below "$rank_i$" is the rank position (according to model result) of the first relevant document (according to human assessors). |Q| denotes the number of queries for which reciprocal ranks summed and averaged over.

$$\mathrm{MRR} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{\mathrm{rank_i}}$$

2.6.2. Baseline

Though this was fundamentally a retrieval task most of the research that tried out CQADupStack has reported their results for duplicate pair classification. As per Doris et al. explain in [34] it is because applying a robust classification model which decides whether a question pair duplicate or not to a full data set takes a longer time. Because generally cqa archives are large in size and iterating over such while comparing against a query question is time consuming. Therefore, researchers use established retrieval algorithms to retrieve a limited candidate set of questions and later apply novel classification algorithms to rerank. Therefore, most report on classification accuracies. My baseline is going to be the result achieved by Di Liang et al. in his research Adaptive Multi-Attention Network Incorporating Answer Information for Duplicate Question Detection [28]. This is the most recent research that can be found which uses CQADupStack for duplicate detection. They claim that the model they proposed can achieve state-of-the-art performance on CQADupStack. But in result tables they compare only the F1 and Accuracy values. Therefore we can assume that what they claim is the correctness of the result rather a performance improvement. Another important thing is that they have given results for individual sub forums and also for CQADupStack full data set. When the domain accuracy is averaged over 12 domains accuracy for one domain it is 0.92 which I'm going to take as the baseline for my research.

On the retrieval side the authors of CQADupStack itself has given a set of baseline results achieved [5] by using TF-IDF, Language Model and also with BM25 algorithm. I'm considering the results of the BM25 model which is the best out of three as the retrieval baseline. They have achieved 0.17 mean average precision for the combined which is the micro average of the results of each sub domain. But the case of using a shared model to evaluate each domain is not experimented for CQADupStack to date. Therefore the results derived in Chen et al. [36] research for multi domain Amazon review data is taken for a rough comparison. They have achieved accuracy varying between 0.85 and 0.91 over 16 domains which has an average of 0.88.

2.8. Summary

According to the literature of the problem there are significantly good achievements so far in each considered area such as Duplicate Detection, Information Retrieval and Domain Adaptation for CQA data. As we reach the modern days of the research in time we can see that the Neural learning and Domain Adaptation is gaining much attention academia and also in the industry especially because of its contribution to image processing domain. In recent years we can see that it also has influenced NLP research as well. In the literature review we discussed a lot of such research that were related to information retrieval, ranking and transfer learning. The essence of most such research is to develop better representations for text.

Vector based representation has become the de facto standard to represent text in most research and the way it derives such vectors are varying in each. Small differences in Neural Layers, Loss functions and also in Hyperparameters has made significant impact on results. We can observe that the Convolutional Neural Networks are better in performance [25] and the Siamese NN which is a Recurrent NN is better in accuracy and also robust with class imbalance data. For relevance ranking similarity scoring function is important and basic distance metrics like cosine, euclidean and manhattan were performing reasonably well and continue to be used in latest NN as well.

Domain adaptation which is also recognized as transfer learning tries to use a model learnet for one domain to be used successfully in another domain for a given task. Therefore the techniques are developed to capture domain invariant features of data while keeping discriminative qualities enough to evaluate relevance against a query. Aggregation of such transfer learning techniques are used to learn common models that satisfies problems over multiple domains. Fundamental technique to add a gradient reversal layer on domain classification sub task to make the final model not discriminative over domains. Adversarial Multi-task Learning [35] is another improved method which is able to reduce the effect of domain specific features and noise when deriving a common model. It's expected to learn a model that gives higher accuracy and efficiency for sub domains than the models learned individually.

When it comes to industry usage of this the nature of the problem is mostly IR. Taking this as a duplicate pair classification task it lets the model to iterate over a very large data set of which all questions are coupled with query questions. But there is a strategy in practice [34] which uses the best classification models to achieve better information retrieval. It applies an industry accepted retrieval algorithm i.e. BM25 or by using an indexing framework [29, 27] like Lucene to initially select a limited number of candidate questions and then do a classification (Duplicate or Not) on that by coupling each with query question. The CQA research to date has focused either duplicate pair classification or retrieval on individual domain. There is a lack in research on duplicate detection for multi domain data focusing a common model.

CHAPTER 3 **METHODOLOGY**

3.1. Overview

Focus of this section is to describe the approach that has been taken in this research to address the problem of duplicate detection in multi domain data. The solution which is proposed is having three major areas of improvement. They are,

- Information retrieval
- Text similarity measurement
- Domain adaptation

According to the literature review it's evident that there was ample successful research on each of the above areas. But the attempts taken to detect duplicates in CQA data have put much focus on either classification of question pairs (duplicate vs non) or IR. They have not given enough attention to Domain adaptation or the way this can be achieved practically in a CQA system. Therefore my work is to build a system to fulfill this gap by detecting the duplicates with higher accuracy. This system is composed of models that are more accurate in duplicate pair classification in individual domains. It also has a model built common to all domains by using domain adaptation techniques.

The final aggregated solution takes an approach that has been successfully practiced in similar kinds of research [27, 29, 34] which makes the methodology much practical one. In a nutshell it's relevance ranking of candidate set of questions retrieved from the question data set with respect to a query question. The approach used in this research is an adopted from Yu et al. [27]. Though they have used it to address a question answering problem the objective was similar such that retrieving the most similar question to a given question from a QA knowledge base. At a high level the proposed solution can be shown in two viewpoints such as a use case for entire system and a component view of the classifier from an engineering point of view.

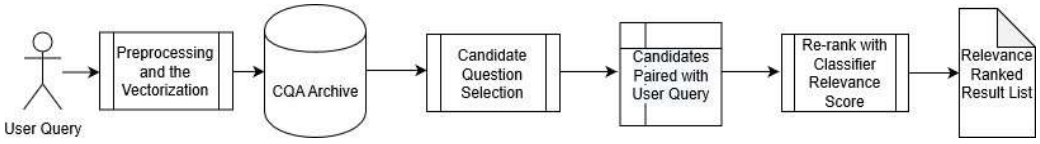

Figure 3.1. Use case of the proposed solution

Figure 3.1 depicts how the solution returns a list of relevance ranked questions based on a given user query. As the diagram shows the user query will be preprocessed and converted to a query vector and then using a standard retrieval mechanism i.e. BM25, Lucene [27] a candidate set of questions will be selected. It can be either equal to the number of questions in the final ranking list or else a higher number. Then we have to construct a data set of question pairs such that each candidate pairs with query questions. Then using a pre trained classifier we will derive the final relevance rankings of candidates against the query.

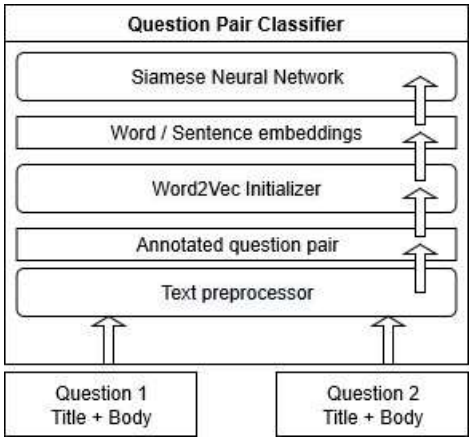

Figure 3.2. Components Diagram of the solution

Figure 3.2. shows how each entity in the classifier of the proposed solution contributes to the duplicate identification. From the components in the Figure 3.2. the classifier gets the main focus of this research as it is the component responsible to improve the Re-rank of Questions. The classifier is generated with a Siamese Neural Network which works well for class imbalanced data such as CQADupStack. The other major area which is improved is the addition of domain adaptation into the equation. This will be applied in the training stage of the Reranking Classifier which is expected to neutralize the domain variance while keeping the ability to

discriminate between duplicates and non. Question Pre-Processing component is there to normalize the varying usage of language and also to take the data into common trainable structure (i.e. sequence length). Application of different embedding implementations such as Word2Vec, Doc2Vec, FastText for question processing is one such attempt which eventually makes a better Classifier.

3.2. Architecture

In this chapter the architecture of Machine Learning Learning solution will be described in detail. It plays the role of question pair classifier for re-ranking which outputs a listing of possible duplicates where the exact duplicate gets a higher rank. Names of each component in the component diagram are self-descriptive. Internal components of this architecture will be discussed along with the intermediate stages of building the Siamese Neural Network in the following subsections of this chapter.

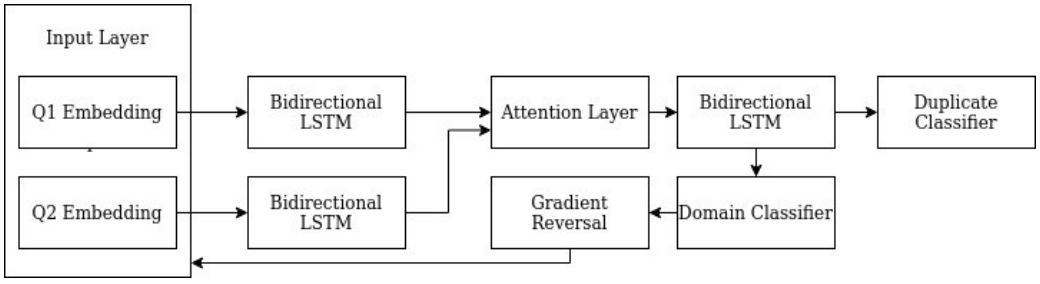

Figure 3.3. Siamese Neural Network

Hyper parameter values of the network

- Embedding dimension = 150
- Maximum sequence length = 110
- Validation split = 0.1
- Rate-drop LSTM = 0.17
- Rate-drop Dense = 0.2
- Number LSTM = 64
- Number of Dense units = 32
- Activation function = 'relu'

3.2.1 Siamese Neural Network

This is a simple recurrent neural network which uses LSTM to learn similarity between two texts. Mueller and Thyagarajan [25] demonstrated that this combination is capable of modeling complex semantics and they used it in a task of sentence similarity learning. Here the word Siamese implies the usage of two recurrent neural networks in parallel. The basic network used in this research is adopted from what proposed by Mueller and Thyagarajan [25] which is shown in the figure 3.5.

3.2.2. Embeddings

This represents the input layer of the Siamese NN which transforms textual inputs into a vector representation of real numbers. Latest methods of embedding generation use neural networks. Two prominent implementations of word embedding generators Word2Vec, FastText are used to derive embeddings specific for the CQADupStack data. These embedding generators do the conversion in three main steps. First it tokenizes the entire document set into word vectors and then trains it to construct word vectors/embedding and finally compose the embedding for each text sequence in the training phase by learning the embedding layer of NN. The alternative mechanism is using a pre-trained language model for which BERT and XLNet are used. More focus was on XLNet [29] as it has recorded best results for most NLP tasks at the moment. The pretrained model is finetuned for the CQADUpStack data and the classification mechanism is adopted from the same implementation which does the finetuning. The results for different Siamese NN combinations are reported in Chapter 4 are based on the Word2Vec embedding of the sequence.

3.2.3. Bidirectional LSTM

The LSTM cells in recurrent neural networks accept words in sentence recursively while keeping historical information in latent space until it reaches the end of sentence before doing an activation to construct an embedding [23]. Bidirectional means connecting hidden layers of such LSTM from both directions to construct the

same output. This way the output layer can get information from both forward (future) and backward (past) states of the hidden layers simultaneously.

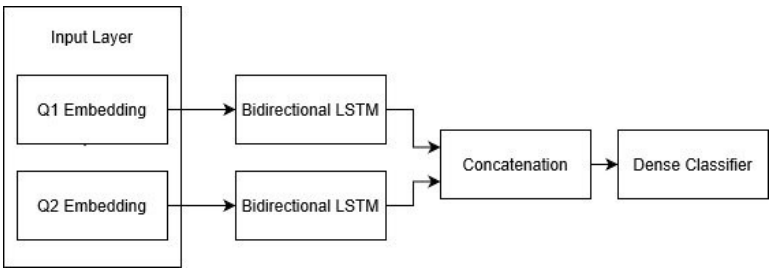

Figure 3.4. Basic version of Siamese Neural Network

3.2.4. Attention Layer

As it's also described in section 2.3.4. the use of the attention layer is to learn much accurate relationships among texts without being contaminated by redundant data that causes hard coupling between different representations in the learning stage. Among the attention mechanisms in literature two are most prominent the Bahdanau attention [37] and Luong attention [38]. Though they are structurally very similar there is a difference in the way they use hidden states to calculate the attention score. And also, Bahdanau attention [37] uses forward and backward source hidden states in the encoder and decoder while Luong uses only hidden states of top LSTM layers. It was the motivation to use an implementation of Bahdanau attention here because the attention between two questions of the same language requires more focus on semantics which can build up from either direction. The version after adding the attention layer is depicted in figure 3.6. This gave a significant improvement in results which can be seen in Figure 3.7.

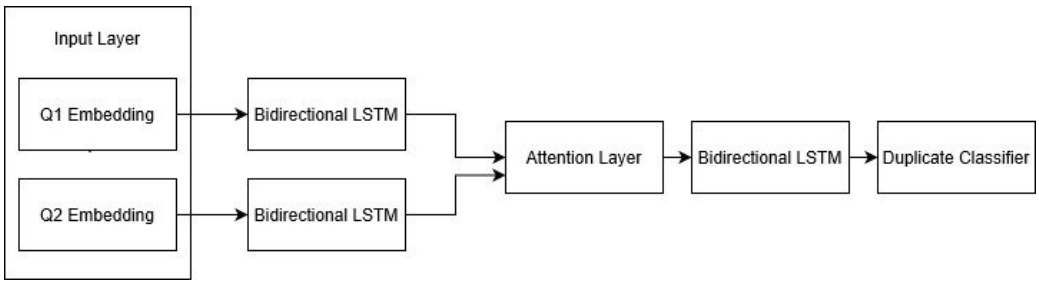

Figure 3.5. Siamese Neural Network with Attention Layer

3.2.5. Adversarial Learning

Gradient Reversal Layer

This was initially proposed by Ganin and Lempitsky [18] for Domain Adaptation where we used an additional classifier for domain determination than the classifier used to achieve the main task. The gradient of the domain classifier is fed to the feature extraction layer inverted by a negative scalar. This is called gradient reversal and we also use the same mechanism in ours. Here it is a classifier which predicts feature vectors into one out of 12 domains of which reversed gradient is feeded back as an input as shown in the figure 3.4. Fundamentals of how the gradient reversal layer connects with the main network is shown in Chapter 2. Figure 2.1.

Multitask Learning

Theoretical background of this approach was discussed in detail in Section 2.4.2. Practically it's an approach to learn models to work better for multiple sub tasks than learning individual models for each. The problem of duplicate detection in multi-domain data is also a multi-task problem. It is having two main tasks such that the identification of duplicate pairs and the domain. The experiment is carried out with a Fully Shared model proposed by But Liu et al. [35]. The theory of Fully Shared model is to share the LSTM layer to learn multiple tasks simultaneously which is shown in figure 2.2. How it has changed the Siamese Network in this research is shown in Figure 3.7.

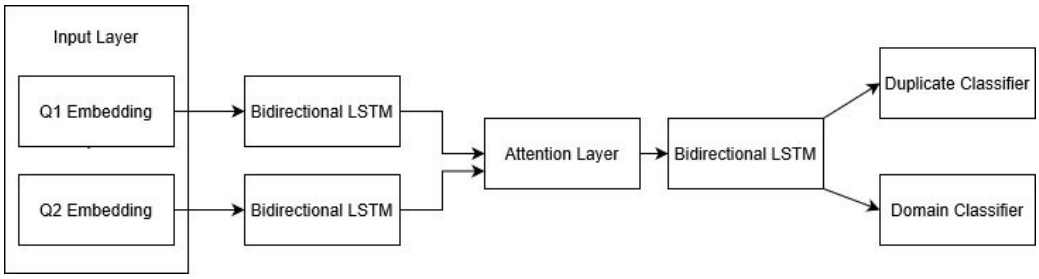

Figure 3.6. Siamese NN for Multitask Learning of Domains

CHAPTER 4 **EVALUATION**

4.1. Introduction

This chapter will showcase the experiment results for classification and retrieval with the configuration being used. Later it will discuss the results and the possible causes for those.

4.2. Classification Results

The classification is mainly for question pairs where the classifier determines whether a pair is duplicate or not. The tests that were carried out for classification targeted the baselines set by Liang et al.[28]. Therefore a data set is also prepared to be compatible with the one they used. The question pairs data set they constructed from CQADupStack was having 1 duplicate pair per 3 question pairs. Few different hyper parameter configurations are tried out in model generation such as changing of vector dimension (150, 300), text sequence length (80, 110) and also the implementation of word embedding generators (Word2Vec, FastText, BERT and XLNet) . In similar configuration settings Word2Vec showed relatively better accuracy than the other two alternatives with much efficient timing for training. Vector dimension as 150, sequence length as 110 for Word2Vec embedding was found to be the best combination and therefore results are generated with those configs. Classification accuracy results for all 12 models are presented in the figure 3.8. that are recorded in different stages of the improvement of siamese architecture. Also, the results of fine-tuned pre-trained mode XLNet is also included in the same graph. Special thing which was observed with XLNet is that it showed better results with shorter text segments (only with title) than the combination of title and body which is opposite to what happened with Siamese LSTM.

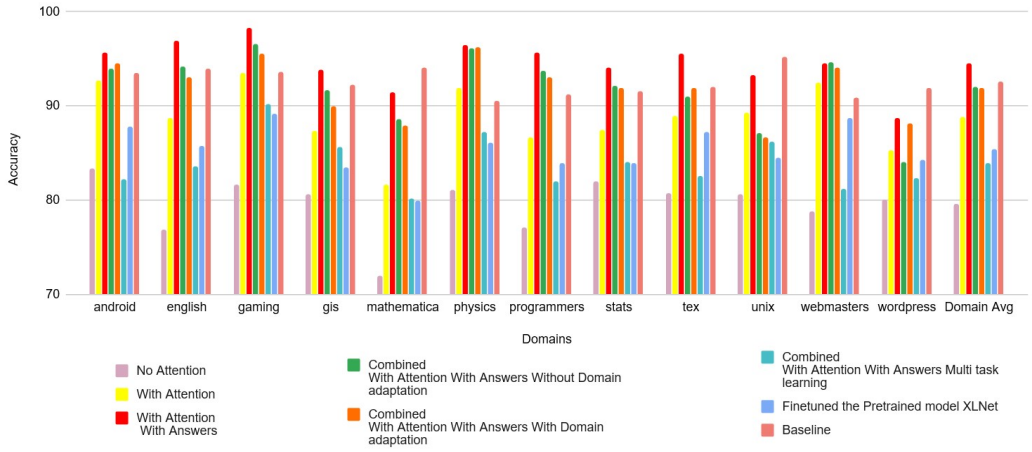

Figure 3.7. Accuracy variation of each domain for different classification strategies

Domain	No Attention	With Attention	With Attention With Answers	Combined With Attention with Answers Without Domain adaptation	Combined With Attention with Answers with Domain adaptation	Combined With Attention with Answers Multitask learning	Finetuned the Pretrained model XLNet	Baseline
android	83.4	92.7	95.66	94	94.51	82.2	87.84	93.44
english	76.9	88.7	96.91	94.16	93.08	83.6	85.7	94
gaming	81.7	93.5	98.33	96.58	95.58	90.2	89.13	93.62
gis	80.6	87.3	93.83	91.67	89.93	85.6	83.51	92.21
mathematica	72	81.6	91.44	88.56	87.94	80.2	79.97	94.01
physics	81.1	91.9	96.5	96.1	96.25	87.2	86.13	90.56
programmers	77.1	86.7	95.6	93.67	92.98	82	83.96	91.18
stats	82	87.5	94.1	92.17	91.91	84	83.98	91.59
tex	80.7	88.9	95.5	91	91.91	82.6	87.2	92.06
unix	80.6	89.3	93.25	87.1	86.66	86.2	84.46	95.23
webmasters	78.8	92.5	94.5	94.6	94.04	81.2	88.68	90.86
wordpress	80	85.3	88.66	84	88.14	82.3	84.25	91.89
Domain Avg	79.58	88.83	94.52	91.97	91.91	83.94	85.4	92.55

Table 4.1. Accuracy values of each domain for different classification strategies

4.3. Retrieval Results

As per my proposed solution I'm expecting the retrieval results of BM25 to be improved after a rerank using the trained classifier. In the retrieval experiment a few different domains are tried out changing the number of candidates for reranking. Here the candidate count was taken as the twice of final selection count. Both are varied from 3 and 6 to 10 and 20 to see the result variation. It can be observed that the re-ranker works better in lower candidate counts and also the precision values getting improved and saturated around result count 10 (20 candidates). Therefore we used a setup which selects 20 candidates and re-rank them to find top 10 as the foundation for retrieval experiments for each domain. Results of that are listed in the figure after.

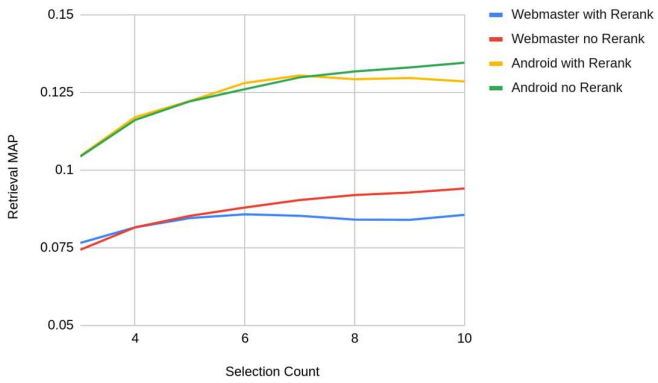

Figure 3.8. Retrieval MAP against the variation of candidate count for rerank

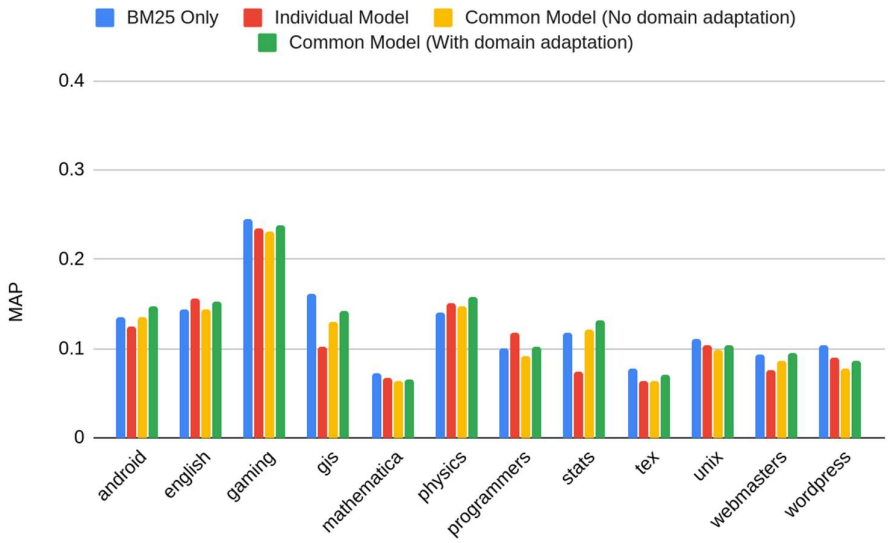

Figure 3.9. Precision (MAP) variation of each domain for different retrieval strategies

	BM25 Only	Individual Model	Common Model (No domain adaptation)	Common Model (With domain adaptation)	Baseline (BM25)
android	0.1346	0.1257	0.1355	0.1483	0.1346
english	0.1442	0.1564	0.1433	0.1522	0.1442
gaming	0.2444	0.2348	0.2313	0.2373	0.2444
gis	0.1619	0.1023	0.1309	0.1415	0.1619
mathematica	0.0724	0.0671	0.0645	0.0662	0.0724
physics	0.141	0.1515	0.1474	0.1588	0.141
programmers	0.1003	0.1182	0.0916	0.1023	0.1003
stats	0.1185	0.0741	0.1221	0.132	0.1185
tex	0.0782	0.0636	0.0647	0.0701	0.0782
unix	0.1113	0.1048	0.0986	0.1033	0.1113
webmasters	0.0941	0.076	0.0864	0.0953	0.0941
wordpress	0.1032	0.0905	0.0783	0.0868	0.1032

Table 4.2. MAP values of each domain for different retrieval strategies

4.4. Discussion

According to the classification results it is evident that the usage of question title + body gives better results than using only the question title. The improvement observed after adding the attention layer is a clear advantage of learning the most important relationships which causes duplication. Classifiers were further improved when using answer texts also in the 2nd question of pairs. This is opposite to what Jiwoon et al. [6] claimed in his research saying that using only the title gives better results. Di Liang et al. [28] has shown that the best results achieved with embedding vectors of dimension 300 but it was not observed a significant improvement in accuracy by varying the dimension beyond 150 with respect to the increased running time. Two types of common models are derived for the combined data such that with or without domain adaptation and they are tested on each domain. The results show that the models trained for individual domains have surpassed the baselines for most of the domains. But the common models have given better accuracy only for domain 'stats'. We have used gradient reversal and multitask learning for domain adaptation out of which only the gradient reversal has given considerable result. But the results show that the domain adapted model also has given nearby accuracy values to individual models. This signals that there is a chance to earn better results for each domain using a common domain adapted model also.

For retrieval using only BM25 has given much better results in some domains relative to what authors have achieved in their baselines [5] with the same technique. But it's hard to assume that the constructed test data set is similar to what they used because of the minimal information provided by the authors. Therefore the BM25 retrieval results earned in the experiment have been taken as the baseline. In the case of 20 candidates where 20 candidates retrieved then reranked and selected top 10. It was expected to gain better MAP with the re-ranker which was achieved by five out of 12 domains compared to the results gained only with BM25. The other special thing is the results obtained by the common model that domain adapted has shown better results than the model for which no domain adaptation is done. It was observed for 10 out of 12 domains.

CHAPTER 5 **CONCLUSION**

Successful text similarity identification and domain adaptation research has already been carried out for community question answering systems. However, it is not evident that a CQA data set which is composed of data from multiple domains has been properly experimented for duplicate detection or retrieval. Therefore, this research was focused on building a better question pair classifier (duplicate or not) and utilizing that classifier to improve duplicate question retrieval.

A Siamese neural network which is composed of two parallel LSTM layers, attention layer and also a dense layer was used as the classifier. Results revealed that the classification accuracy has improved for the domain specific models generated comparative to the most recent baseline [28] which was selected. It was able to surpass baselines for 9 out of 12 domains. A model using data from all the domains is also trained as a common evaluator for each domain. The versions which included a gradient reversal layer and the version with multitask learning was also used as common models. But these models (with or without domain adaptation) have not shown significant improvement against the baseline. The next experiment was utilization of the generated models to improve the recall rate and precision of retrieval. Authors of CQADupStack reported BM25 as the baseline. The approach taken in this research used BM25 to retrieve candidates and then re-rank with trained classifiers. This approach was able to surpass baselines for 6 out of 12 domains. The important observation here is the success of domain adapted version over the others.

With the results and observations discussed this research can be concluded as proof that attention based Siamese neural networks which are domain adapted can be successfully used to improve the CQA duplicate retrieval. Also, the attention based CQA question pair classifiers can be successfully trained to achieve better accuracy by including title body and the answers of questions while training. This research will be beneficial for community forums and for the industry to implement better automated duplicate question detection strategies.

There are areas to investigate further in this research specially to improve recall rates in retrieval. Novel embedding and domain adaptation techniques can be used for this purpose. Text sequence and embedding sizes of higher order which are usually processing resource greedy can be further tried out for different combinations with better computation power. There are avenues of future research that are slightly deviating from the focus of this research such as how to make the existing techniques to work with less computation power for higher order sequence and embedding dimensions.

REFERENCES

[1] J. Bian, Y. Liu, E. Agichtein, and H. Zha, "Finding the right facts in the crowd," *Proceeding of the 17th international conference on World Wide Web - WWW '08*. 2008.

[2] X. Wang, L. Zhang, T. Xie, J. Anvik, and J. Sun, "An approach to detecting duplicate bug reports using natural language and execution information," *Proceedings of the 13th international conference on Software engineering - ICSE '08*. 2008.

[3] H. Wang and P. Poupart, "Overfitting at SemEval-2016 Task 3: Detecting Semantically Similar Questions in Community Question Answering Forums with Word Embeddings," *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. 2016, doi: 10.18653/v1/s16-1133.

[4] P. Runeson, M. Alexandersson, and O. Nyholm, "Detection of Duplicate Defect Reports Using Natural Language Processing," *29th International Conference on Software Engineering (ICSE'07)*. 2007.

[5] D. Hoogeveen, K. M. Verspoor, and T. Baldwin, "CQADupStack," *Proceedings of the 20th Australasian Document Computing Symposium on ZZZ - ADCS '15*. 2015.

[6] J. Jeon, W. Bruce Croft, and J. H. Lee, "Finding similar questions in large question and answer archives," *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM '05*. 2005.

[7] H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu, "Searching Questions by Identifying Question Topic and Question Focus," in *ACL*, 2008, pp. 156–164.

[8] M. Franco-Salvador, S. Kar, T. Solorio, and P. Rosso, "UH-PRHLT at SemEval-2016 Task 3: Combining Lexical and Semantic-based Features for Community Question Answering," *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. 2016, doi: 10.18653/v1/s16-1126.

[9] A. Singh, "Entity Based Q&A Retrieval," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, 2012, pp. 1266–1277.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv [cs.CL]*, Jan. 2013.

[11] K. Wang, Z. Ming, and T.-S. Chua, "A syntactic tree matching approach to finding similar questions in community-based qa services," *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information*

retrieval - SIGIR '09. 2009.

[12] H. Yang and J. Callan, "Near-duplicate detection by instance-level constrained clustering," *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '06*. 2006.

[13] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[14] C. J. Burges, R. Ragno, and Q. V. Le, "Learning to Rank with Nonsmooth Cost Functions," in *Advances in Neural Information Processing Systems 19*, P. B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 193–200.

[15] J. Guo *et al.*, "A Deep Look into neural ranking models for information retrieval," *Information Processing & Management*. p. 102067, 2019.

[16] T.-Y. Liu, "Learning to Rank for Information Retrieval." 2011, doi: 10.1007/978-3-642-14267-3.

[17] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," Nov. 1999.

[18] Y. Ganin and V. Lempitsky, "Unsupervised Domain Adaptation by Backpropagation," *arXiv [stat.ML]*, 26-Sep-2014.

[19] I. Goodfellow *et al.*, "Generative Adversarial Nets," in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2014, pp. 2672–2680.

[20] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep Domain Confusion: Maximizing for Domain Invariance," *arXiv [cs.CV]*, 10-Dec-2014.

[21] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial Discriminative Domain Adaptation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[22] X. Qiu and X. Huang, "Convolutional neural tensor network architecture for community-based question answering," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[23] H. Palangi *et al.*, "Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 4. pp. 694–707, 2016.

[24] Z. Wang, W. Hamza, and R. Florian, "Bilateral Multi-Perspective Matching for Natural Language Sentences," *Proceedings of the Twenty-Sixth International Joint*

Conference on Artificial Intelligence. 2017.
[25] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
[26] S. Romeo *et al.*, "Neural attention for learning to rank questions in community question answering," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 1734–1745.
[27] J. Yu *et al.*, "Modelling Domain Relationships for Transfer Learning on Retrieval-based Question Answering Systems in E-commerce," *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM '18*. 2018.
[28] D. Liang *et al.*, "Adaptive Multi-Attention Network Incorporating Answer Information for Duplicate Question Detection," *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR'19*. 2019.
[29] D. Hoogeveen, L. Wang, T. Baldwin, and K. M. Verspoor, "Web Forum Retrieval and Text Analytics: A Survey," *Foundations and Trends® in Information Retrieval*, vol. 12, no. 1. pp. 1–163, 2018.
[30] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial Multiple Source Domain Adaptation," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 8559–8570.
[31] D. Cohen, B. Mitra, K. Hofmann, and W. Bruce Croft, "Cross Domain Regularization for Neural Ranking Models using Adversarial Learning," *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval - SIGIR '18*. 2018.
[32] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
[33] S. Robertson, "The Probabilistic Relevance Framework: BM25 and Beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4. pp. 333–389, 2010.
[34] D. Hoogeveen, "Real and misflagged duplicate question detec-tion in community question-answering," PhD thesis.

[35] P. Liu, X. Qiu, and X. Huang, "Adversarial Multi-task Learning for Text Classification," *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2017, doi: 10.18653/v1/p17-1001.

[36] X. Chen and C. Cardie, "Multinomial Adversarial Networks for Multi-Domain Text Classification," *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, doi: 10.18653/v1/n18-1111.

[37] D. Bahdanau, K. Cho, and Y. Bengio. "Neural machine translation by jointly learning to align and translate," *In Proceedings of the International Conference on Learning Representations (ICLR). 2015*

[38] Luong, Minh-Thang, Hieu Pham, and Christopher D. Manning. "Effective Approaches to Attention-based Neural Machine Translation," *arXiv preprint arXiv:1508.04025* (2015).

[39] J. Devlin, MW Chang, K Lee, K Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805. 2018 Oct 11.

[40] Z Yang, Z Dai, Y Yang, J Carbonell, RR Salakhutdinov, Le QV. "Xlnet: Generalized autoregressive pre training for language understanding." *In Advances in neural information processing systems 2019 (pp. 5754-5764).*

[41] Miller GA. "WordNet: a lexical database for English." *Communications of the ACM.* 1995 Nov 1;38(11):39-41.