Project Report

Dipam Paul

Generating arXiv citation graph with Recurrent Neural Networks

Nov 2020

Abstract

Presently, arXiv has a rich collection of over 1.6 million electronic preprints of scientific articles in various fields, which can be accessed online. With such a huge collection of research papers, arXiv has become a primary source of literature for researchers. However, the usefulness of a bibliographic database depends highly on a consistent citation graph which links the papers present in the database. Owing to the large variations in citation formats used for listing bibliographic information, it has become an increasingly difficult problem to accurately find the corresponding cited papers. In this paper, we make an attempt towards recreating arXiv citation graph using only the raw LaTeX dump of research articles. We present a novel generalised pipeline which employs a combination of traditional heuristics and modern deep learning methods to find the list of papers cited by an article. We evaluate the results of the pipeline on the datasets provided in the data cleaning task of the KDD Cup 2003 which comprises a collection of over 35,000 articles submitted in the hep-ph (High Energy Physics - Phenomenology) section of arXiv between 1992 and 2003. Our pipeline extracts around 285k edges present in the hep-ph citation graph correctly out of 421k edges present in the true citation graph. This achieves an accuracy of around 95% while matching citations to the corresponding article and outperforms the top solutions presented in KDD Cup 2003 by a large margin with a 64% relative improvement in symmetric difference size. To validate the generality of the approach, we also report empirical results of the pipeline over another dataset presented in the same challenge for citation prediction task, which contains around 29,000 hep-th (High Energy Physics - Theory) papers submitted on arXiv.

Contents

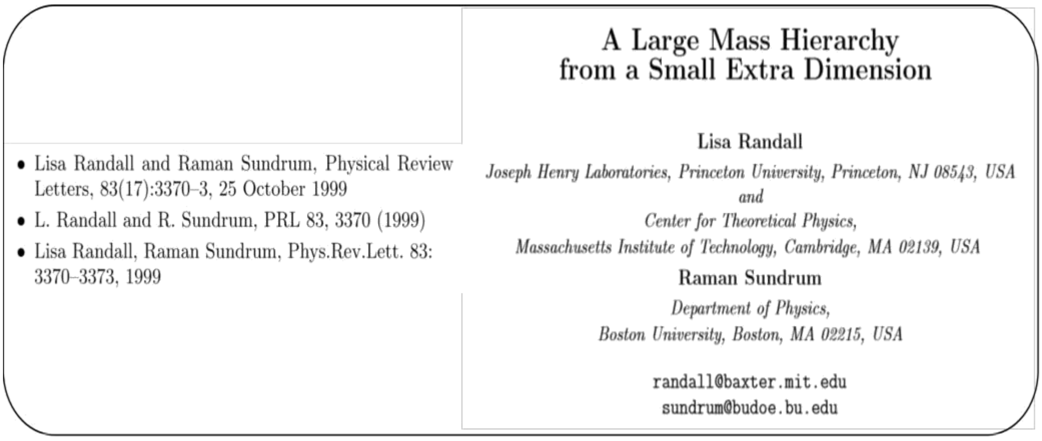
Chapter 1

Introduction

Modern bibliographic databases provide the basis for scientific research and its evaluation. The prime example of a bibliographic database is arXiv.org. arXiv[1] has become the primary mode of research communication in multiple fields of physics, and some related disciplines. It contains over 1.6 million full-text articles in the fields of physics, mathematics, computer science, quantitative biology, quantitative finance, statistics, electrical engineering and systems science, and economics, and is growing at a rate of 120k new submissions per year. Such a huge collection of scientific literature opens up many possibilities for progressing, analyzing and evaluating the research in any field.

The usefulness of a bibliographic database such as arXiv depends highly on a consistent citation graph which links the papers. The graph provides huge clues as to which article is similar to other. Previous methods by SLAC/SPIRES [1] employ automated heuristics followed by human post-processing to find the papers cited by an article. The automated portion of this process is being refined over many years using extensive domain knowledge. However, there are still many cases in which the process fails to perform well. Since researchers use a myriad of different citation formats, finding the proper corresponding paper to a given citation and in general re-constructing the arXiv citation graph poses a highly non-trivial challenge. Moreover, the actual bibliography present in the papers is not clean, i.e., citations contain spelling variations on author names, abbreviations, typos, and other sources of noise. A trivial example of differences in citations referring to the same paper [1] [2] is shown in Figure 1.1. However, more obscure examples in

[1] https://arxiv.org



- Lisa Randall and Raman Sundrum, Physical Review Letters, 83(17):3370–3, 25 October 1999
- L. Randall and R. Sundrum, PRL 83, 3370 (1999)
- Lisa Randall, Raman Sundrum, Phys.Rev.Lett. 83: 3370–3373, 1999

A Large Mass Hierarchy
from a Small Extra Dimension

Lisa Randall
Joseph Henry Laboratories, Princeton University, Princeton, NJ 08543, USA
and
Center for Theoretical Physics,
Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Raman Sundrum
Department of Physics,
Boston University, Boston, MA 02215, USA

randall@baxter.mit.edu
sundrum@budoe.bu.edu

Figure 1.1: Different citations referring to the same research paper

which a part of the information is missing (i.e., author names, title, journal or year of publication) also exist in practice. For example, the citation listed below doesn't include the name of the journal in which the text is published - “J. Cleyman, K. Redlich, H. Satz, and E. Suhonen, 1993.”

In addition to these problems, a fair amount of the papers on arXiv have published duplicates [3]. The fact that some of the arXiv papers cite published duplicates instead of the articles in its database strongly hinders the formation of a consistent in-corpus citation graph. However, the success of deep learning in processing and analyzing texts provides a huge promise to make some advances to solve this problem.

In this work, we focus on tackling the problem of creating a consistent citation graph of scientific articles present on arXiv using modern deep learning (DL) techniques. We reduce citation graph construction into familiar NLP tasks such as named entity recognition and finding semantic text similarity. We demonstrate a DL-based approach to tackle this problem and thus, create a huge citation graph of research articles as shown in Figure 1.2. The nodes of the graph represent research papers and directed edges from a node indicate the articles cited by the paper represented by that node. The graph can be thought of encoding an implicit relationship between papers - in terms of authorship and the transmission of ideas. Moreover, such a huge network of citations forms a more structured representation of the space of research, thus making it easier to design powerful graph-based algorithms and queries for analyzing and obtaining useful statistics and insights [4] [5] [6] about the papers present in a particular field of research. To the best of our knowledge, this is the first work in the direction of creating an end-to-end

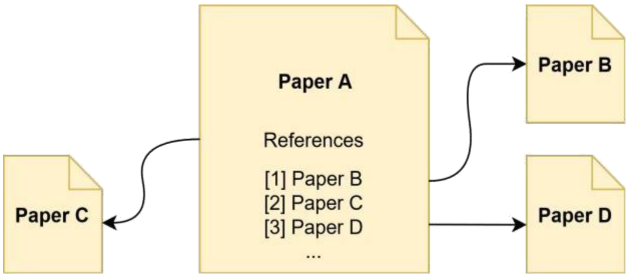


Figure 1.2: Citation graph

pipeline for citation graph construction with comparatively significant gains.
The contributions of our work are -

We present a 3-stage pipelined approach[2] which employs a combination of heuristics and modern deep learning techniques to find the list of papers cited by an article. The approach first extracts citations from the bibliography section of the article, followed by a step to parse useful entities such as author names, title, journal, year from the citations. This structured information is then used to match these citations to the respective articles and return a list of articles cited by the paper.

We demonstrate using examples that the approach is robust to missing information and other sources of noise in the citations.

The pipeline achieves significant advances while reconstructing the citation graph of articles present in hep-ph section of arXiv using the dataset presented in KDD Cup 2003.

We also evaluate the potential of the pipeline to generalize to other sections of arXiv. Specifically, we present the results of the same pipeline (without re-training the models) on hep-th section of arXiv.

The remainder of the paper is arranged as follows. Chapter 2 elucidates how the problem of extracting information from a rich collection of scientific literature has progressed over the years and sheds some light on some prominent solutions that have been developed. Additionally, it briefly reviews various rule-based and machine learning-based approaches which addressed the problem of creating a consistent citation graph of research articles. Chapter 3 deals with a comprehensive description of each of the components of our 3-stage pipeline. It presents the architecture of the machine learning models used. In chapter 4, we provide an overview of the datasets used for evaluating the proposed approach. In chapter 5, we provide a detailed discussion of the experiments we ran followed by the results of the pipeline. Chapter 6 concludes and summarizes the paper.

Chapter 2

Related Work

There has been a tremendous growth of open-access scientific literature on arXiv.org over the last decade. The vast amount of literature available forms the basis for analyzing and promoting new frontiers in scientific research. However, at the heart of such a good bibliographic database is the knowledge graph of citations present in the articles within the database. Parsing relevant information from research articles for performing various useful tasks has been an active area of work over the last few decades. There has been a tremendous work in this direction so as to facilitate an easy access of right information to the researchers. These approaches can primarily be divided into two categories - heuristics/rule-based approaches and machine learning-based approaches. This chapter briefly reviews some of the prominent approaches developed so far and also focuses on some advancements in the direction of constructing a citation graph.

2.1 KDD Cup 2003 - Data Cleaning Task

This problem of arXiv citation graph construction was previously addressed in the Data Cleaning Task of the KDD Cup 2003 [2]. The goal was to recreate the citation graph for papers from a category called hep-ph, using only the LaTeX sources of the papers. The problem turned out to be quite difficult due to the unclean LaTeX sources and a wide variety of different styles used for writing the documents. The winning approaches exploited a combination of meticulously handcrafted features [1] [7] [8] to find the best mapping between citations and papers. Nonetheless, the best solution [7] was able to extract only 40,600 edges out of 421k present in the true citation graph. The results were not so satisfactory given that the prediction accuracy of the algorithm was only around 23% and this can be attributed to the mediocre performance of the information extraction algorithm which was based on regular expression matching.

2.2 Rule-based Approaches

Rule-based approaches employ a combination of logic and regular expression-based matching to extract information from articles. [9] proposed an algorithm for extracting meta-data from PostScript sources of scientific papers using a set of rules based on spatial and visual properties of texts. PDFX [10] reconstructs the logical structure present in PDF sources of papers in the form of a parsable XML document from which the references section can be extracted. Another prominent example of a rule-based system is pdf-extract [11], an open source tool by Crossref Labs for extracting citation references (and, eventually, other semantic metadata) from PDFs. It uses a combination of visual cues and content peculiarities to identify semantically significant regions of the PDF such as the references section and finally extracts individual references. Refextract [12] by INSPIRE-HEP is also an open-source tool, which is designed to extract bibliographic references specifically from High-Energy Physics research papers using a set of carefully designed rules. [13] [14] proposed a template-mining based approach for creating a citation database. [15] [16] explored a knowledge-based approach for extracting references. A major downside of such rule-based methods is that they are not robust to the variations and noise present in reference styles. In fact, in practice, the set of regular expressions and logical programs needs a constant update to adapt to variations in new data.

2.3 Machine Learning-based Approaches

Many machine learning algorithms [17] have been proposed to solve the problem of reference parsing from scientific articles. [18] [19] reported the use of Hidden Markov models [20] for extracting metadata from citations while [21] [22] proposed a SVM-based approach for the task. [23] presented a two-step approach based on statistical machine learning techniques to extract different entities in references, such as author names, title, journal name and publication date. They formulated reference localisation as a classification problem based on various text and geometric features, and employed algorithms like conditional random field and support vector machine to parse entities from each reference. [24] also considered reference parsing as a sequence labelling problem and explored a SVM based algorithm for this task. Both these approaches achieved near-perfect accuracy for parsing references, but these methods have been evaluated on a very small dataset consisting of only 500 samples. The dataset which we use is larger and has much larger inherent complexity. Similar is the case with other works discussed below.

Another prominent work is CERMINE [25] which achieves an average F score of 77.5% for extracting structured metadata from PDF sources of scientific articles. Moreover, [25] reported the ability of CERMINE to adapt to new document styles and formats owing to the use of supervised and unsupervised machine learning techniques. [26] [27] proposed GROBID which uses Conditional Random Fields (CRF) [28] in order to extract various information from PDF source of an article such as metadata, full article text and a list of parsed bibliographic citations. ParsCit [29] also uses CRF for parsing reference strings. SectLabel [30] uses CRF with features extracted from the documents using Optical Character Recognition to locate bibliography section in a research paper. Furthermore, Neural ParsCit [31], a recent work developed on top of ParsCit, exploits a Long short-term memory (LSTM) [32] based network to parse string references. [33] presented a tool called FLUX-CiM, which relies on an automatically constructed knowledge base for extracting components from references.

In the recent years, there have been several notable attempts to create a huge citation graph of scientific literature. For instance, CiteSeerX [34] (previously CiteSeer [35]) is a digital library search engine which uses machine learning techniques for accomplishing several tasks essential to the working of the engine. These techniques accurately extract metadata, cluster documents and citations and also try to tackle the problem of author disambiguation to some extent. Moreover, CiteSeerX also uses ParsCit [29] at its core for extracting and parsing citations. [36] proposed a system for producing a symbolic graph representation of published scientific literature on www.semanticscholar.org. They experimented with RNN-based bidirectional language models for extracting information from raw PDF sources and thus, constructed a literature graph consisting of more than 280M nodes, representing papers, authors, entities and various interactions between them. A large contextual citation graph of 81.1M academic publications was released by [37] out of which bibliography entries could be parsed and linked from only 27.6M articles, leading to 380.5M citation edges. The graph was constructed by using GROBID [27] to extract relevant information from the PDF sources of papers. Additionally, [3] also built a large co-citation network of arXiv articles with 6.7 million edges, but their work majorly relied on the use of arXiv identifiers which makes the task much easier.

Chapter 3

Methodology

3.1 Introduction

In this chapter, we introduce the proposed approach for creating a citation graph of research articles. There are three major steps in our pipelined approach as shown in Figure 3.1.

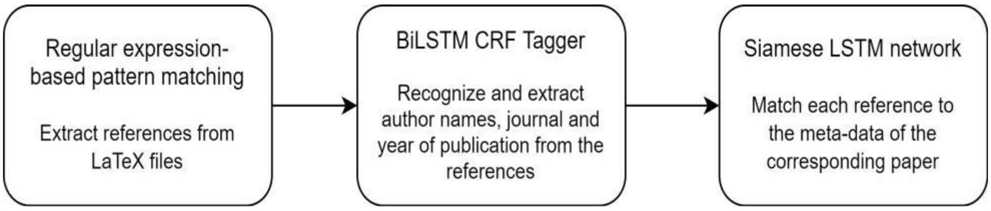

Regular expression-based pattern matching
Extract references from LaTeX files
BiLSTM CRF Tagger
Recognize and extract author names, journal and year of publication from the references
Siamese LSTM network
Match each reference to the meta-data of the corresponding paper

Figure 3.1: Overview of three-stage pipeline

First, given the LaTeX source of an article, we extract the references from the file using regular expression-based (regex) pattern matching. Secondly, we parse relevant information from these citations in the form of author names, journal name, title and year of publication using a custom Bi-LSTM CRF-based named entity recognition network. Finally, we used this information to match the citation to the corresponding paper using a two-level hierarchical matching based on Siamese LSTM network (Manhattan LSTM). An example of the workflow is shown in Figure 3.2.

A brief description of each of the component of the 3-stage pipeline and its detailed implementation is provided in this chapter.

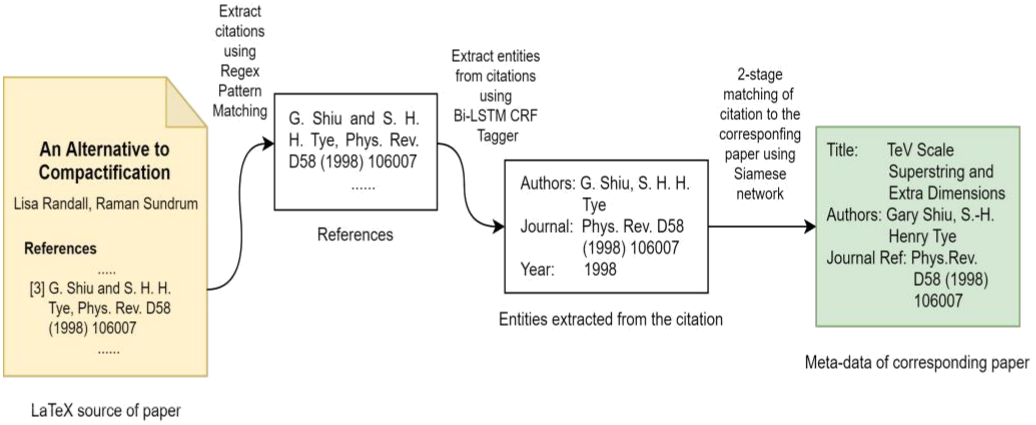

An Alternative to Compactification
Lisa Randall, Raman Sundrum
References
.....
[3] G. Shiu and S. H. H. Tye, Phys. Rev. D58 (1998) 106007
......
LaTeX source of paper
Extract citations using Regex Pattern Matching
G. Shiu and S. H. H. Tye, Phys. Rev. D58 (1998) 106007
......
References
Extract entities from citations using Bi-LSTM CRF Tagger
Authors: G. Shiu, S. H. H. Tye
Journal: Phys. Rev. D58 (1998) 106007
Year: 1998
Entities extracted from the citation
2-stage matching of citation to the corresponfing paper using Siamese network
Title: TeV Scale Superstring and Extra Dimensions
Authors: Gary Shiu, S.-H. Henry Tye
Journal Ref: Phys.Rev. D58 (1998) 106007
Meta-data of corresponding paper

Figure 3.2: Example workflow of the pipeline

3.2 Extracting References using Regular Expressions

The extraction of citations from the LaTeX source of a research paper involves determining the part of text where the references are situated. Unlike [27] [29], we employ a simple heuristic-based regular expression (regex) pattern matching algorithm to extract citations from the bibliography section of the article.

In order to ensure high precision and robustness while identifying references, we believe that it is essential to divide the problem in two distinct tasks. First, the algorithm localizes the bibliography or reference section in the LaTeX file by locating predefined patterns which may correspond to such a section. We carefully derived a number of regular expressions capable of dealing with the nuances in writing formats and styles while identifying the bibliography section. If the algorithm succeeds in locating the bibliography section, we need to locate and extract the citations from the bibliography section. This is quite straightforward and is done by looking for identifiers which usually separate citations or identify the beginning of new citation (for example, "\bibitem", or "[NUMBER])". Finally, we clean the identified citations using obvious steps such as removing common LaTeX commands, special characters and punctuation (except dot, comma and semicolon as these prove to be helpful as evident in the later sections).

3.3 Extracting Information using Bi-GRU CRF Tagger

Once the references in the scientific text have been identified and localised, the next step involves extracting key information say, author names, journal reference, title and year of publication from these citations. We view the problem of extracting these fields as a traditional named entity recognition problem. Deep recurrent neural networks have been shown to be effective [38] [39] [40] for such sequence labelling tasks. So, we exploit bi-directional recurrent neural networks (Bi-RNN) combined with conditional random field (CRF) [28] [38] to extract relevant fields from the citation. The network takes a citation as input and labels each word in the citation with one of the possible tags - author (A), title (T), journal (J), year of publication (Y) and other (O). For the RNN layer, we specifically employ gated recurrent unit (GRU) layers due to their computational efficiency and the fact that citations are usually shorter in number of words. The overall architecture of the network is shown in Figure 3.3.

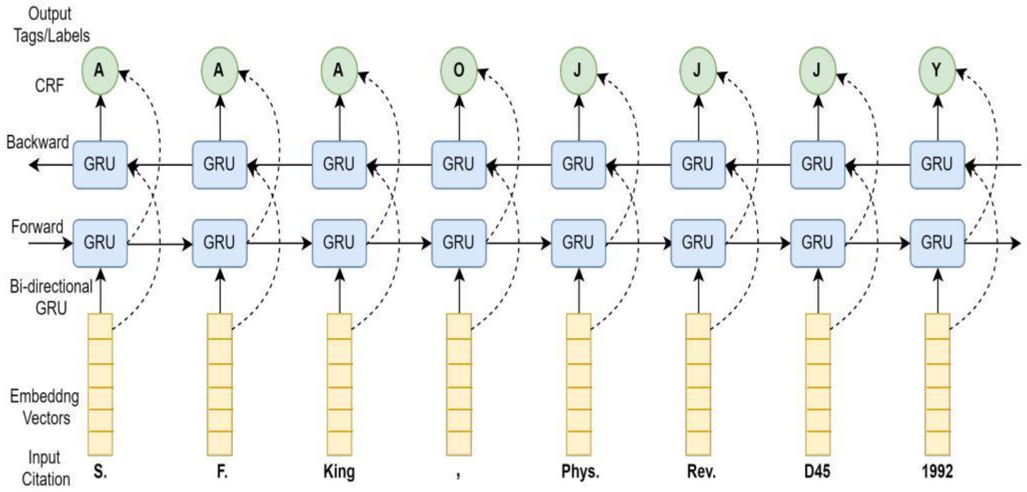


Figure 3.3: Architecture of Bi-GRU CRF network for parsing information from citation

The benefit of the BiGRU-CRF architecture is that it can use both past and future input features due to the bidirectional GRU layer. In addition, the CRF layer enables the model to use neighbouring tag information. Intuitively, it considers the correlation between tags in the neighborhood of a word and jointly decodes the best sequence of tags for a given input citation text.

Suppose the input citation is represented as x_{cite} = [x_1; x_2; : : : ; x_{Nc}], the sequence of word-labels for the citation is y = [y_1; y_2; : : : ; y_{Nc}] and there are K possible tags. Let g(x_{cite}) be the encoded representation of the input citation produced by the bi-directional GRU layer and s_i = n(g(x_{cite})) represent the probability output by the Bi-GRU network for each label assigned to the current word x_i with s_i 2 R^K . Here, n is a linear mapping.

We define a score function (x_{cite}; y) which represents how well the label sequence y fits the given input sequence x_{cite} and is expressed as:

$$(x_{cite}; y) = \underset{=1}{\overset{N_C}{x}} s_i(y_i) + T_{y_i \ 1 \ \ ! y_i} \qquad (3.1)$$

where, $T_{y_i\ 1\ !\ y_i}$ is the weight corresponding to the transition from label y_i $_1$ to label y_i

Then, the CRF models the conditional probability represented by (3.2).

$$\begin{aligned} p(y j x_{cite}) &= p(y_1; y_2; : : : ; y_{Nc}\ j x_1; x_2; : : : ; x_{Nc}) \\ &= P\frac{\exp(\ (x_{cite}; y))}{y_0 \exp(\ (x_{cite}; y^0))} \end{aligned} \qquad (3.2)$$

This allows the network to use the information from neighbouring labels while predicting the label for the current word and performs better than directly using the outputs of Bi-GRU layer and enables the network to make global choices.

Because the use of Bi-RNN CRF networks for such tasks has become common and our model architecture is quite simple, we omit an exhaustive background description of other layers of the network and instead focus on the details of training the network later in chapter 5. We use pre-trained GloVe embedding [41] of the highest dimensions (300D) and largest vocabulary, to embed the input citation and then feed these vectors to the Bi-GRU layer word by word. For words which are not present in GloVe vocabulary, we initialize the word vector randomly and learn the vector while training the model. This seems essential as the vector may be a person (author) name which is not present in GloVe vocabulary, but is of vital importance for our task.

3.4 Siamese LSTM Network for Matching Citation to the Corresponding Paper

The last stage of the pipeline is to match or link each citation to the corresponding research article using all the information we have extracted so far. We employ a variant of Siamese LSTM network [42] to perform the matching task. It is based on the principle that deals with transforming input vectors to a space where it is easier to gauge the similarity between them. Let x_{cite} be the citation which we want to match and x_{meta} be the meta-data of some arbitrary paper. Then, the network consists of twin LSTM-based [32] subnetworks, say M and C, which accept two distinct inputs x_{meta} and x_{cite}. Suppose the subnetwork M encodes the input sequence x_{meta} from input space of dimensions d_{in} to a target space of dimensions d_{out} and the subnetwork C encodes the input sequence x_{cite} from dimensions d_{in} to d_{out}. The main idea is to approximate a suitable function using the subnetworks M and L, that maps input text into a space so that a simple distance in the target space approximates the "semantic" distance between the two sequences in the input space. The function f then computes a measure of similarity between the highest-level feature representations produced by the respective subnetworks. Thus, the output of the network provides an estimate of the similarity between the meta-data and the citation.

An essential trick for the working of this is that the parameters between the two subnetworks are tied i.e., same weights are used in both the networks. Weight tying guarantees that two similar inputs cannot possibly be mapped by their respective subnetworks to very different points in feature space because each network essentially computes the same mapping. Traditionally [42] [43], semantic similarity between the output representations produced by the twin subnetworks is computed by a simple L1- norm followed by sigmoid non-linearity. We believe that a better approach would be to instead use a fully connected layer at the top and let the distance function f be approximated by the layer. The two Bi-LSTM encoders each, will encode the semantics present in the input into fixed-dimensional hidden vectors.

If z_{meta}^0 and z_{cite}^0 are the representations of the inputs produced by the respective Bi-LSTM subnetworks, these vectors can be compared by the

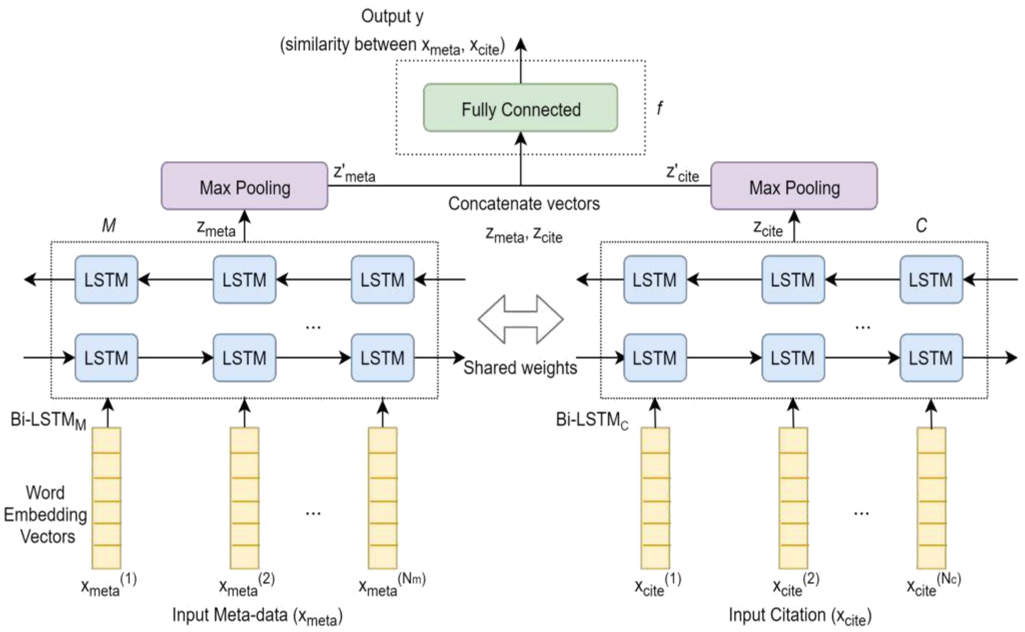


Figure 3.4: Siamese LSTM network architecture for matching citations to corresponding papers

fully connected layer to estimate the output which is formulated as in (3.3).

$$
\begin{aligned}
y &= f(z_{meta}^{0}; z_{cite}^{0})) \\
&= 1; \text{ if citation matches the meta-data (target paper is found)} \\
&= 0; \text{ if citation does not match the meta-data}
\end{aligned}
\tag{3.3}
$$

Thus, the resulting architecture which we use is illustrated in Figure 3.4.

For training the network, we provide pairs of meta-data and citation along with a label for the pair which is 1 only when the citation and meta-data correspond to the same paper. Furthermore, we used cross-entropy objective as it seems a natural choice for training the network.

Chapter 4

Dataset

In this chapter, we provide a brief overview of the datasets used for evaluating the performance of the approach described in this work. We use the dataset provided in the data cleaning task of the KDD Cup 2003 which consists of unprocessed LaTeX sources of 35,866 papers published in hep-ph section of arXiv until March 2003. The dataset is divided into 11 gzip files with 3000-4000 papers in each file as shown in Figure 4.1. For each paper, the dataset only includes the main .tex file and does not include any other files or figures. Also, a small subset of papers are present as plain text files as LaTeX files were not available for these papers. We, specifically, choose to work with this dataset due to the inherent difficulty, noise and variations present in the dataset. The citations present in the texts are unclean with spelling variations on author names, abbreviations, typos, and other sources of noise. Moreover, citations in the physics community usually do not contain the paper title, which makes the task of matching even more strenuous. The true citation graph of the articles released by KDD after the challenge ended, has about 421,000 edges between these 35,000 papers. The graph has been compiled by SLAC/SPIRES using automated heuristics followed by human post-processing. The automated heuristic algorithm has been refined over many years and uses arXiv identifiers to construct the graph. However, to make the task challenging, all unique arXiv identifiers have been stripped off from these LaTeX sources.

To assess the generality of the approach, we also run the proposed pipeline over the dataset presented in the citation prediction task of KDD Cup 2003. It contains LaTeX sources of over 29,000 papers submitted in hep-th portion of arXiv until May 2003. This data is divided into 12 parts according to the year as shown in Figure 4.2. In order to be consistent, we removed the arXiv identifiers from these files as well. The true citation

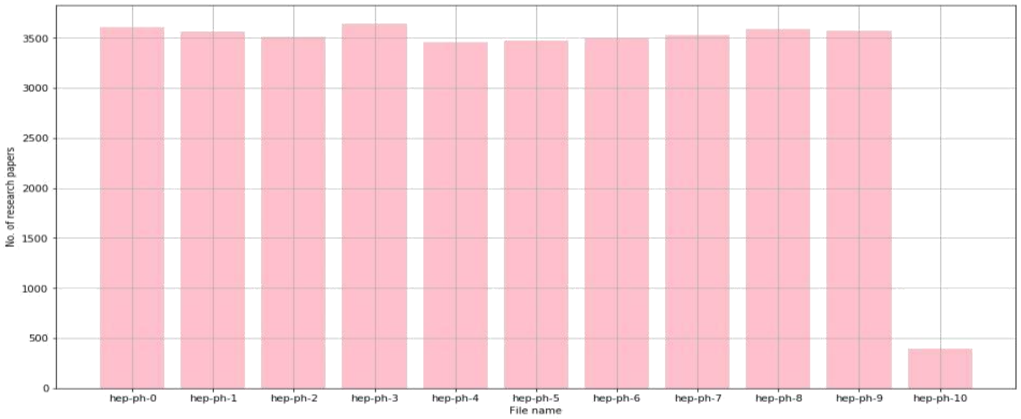


Figure 4.1: Number of research papers in each file for hep-ph dataset

network corresponding to this dataset, has about 350,000 edges between these papers.

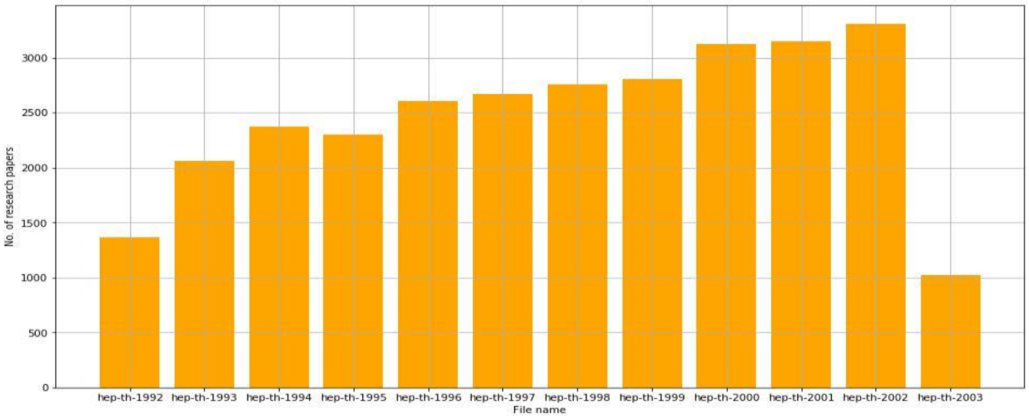


Figure 4.2: Number of research papers in each file for hep-th dataset

Other than these two data sources, we use the meta-data of articles provided by arXiv to construct a clean, reliable, noise-free corpus for supervised training of our models - namely, BiLSTM CRF tagger and Manhattan LSTM network. We use the code provided by [3] which uses OAI protocol for bulk arXiv metadata download. We thus obtain a full set of meta-data for hep-ph and hep-th articles and filter the set to only contain meta-data of articles submitted between the years 1992 and 2003 on arXiv. The meta-data for each paper includes an arXiv ID, a list of author names, title, abstract, versions, category listings, journal reference and report number fields. There are entries in which either of the fields is

Field	arXiv ID	Authors	Title	Journal	Report No	Year
Count	48329	48329	48329	35058	26650	48329

Table 4.1: Count of non-null fields in meta-data

missing. Table 4.1 shows the counts of meta-data entries in which the corresponding field is present.

Chapter 5

Experiments and Discussion

In this chapter, we present the results of the 3-stage pipeline on the datasets provided in KDD Cup 2003 and investigate the performance of the approach. We also provide detailed ablation studies to understand the effect of various design decisions on the overall performance of the pipeline. However, due to the lack/unavailability of strong baselines, it was not possible to provide a detailed empirical comparison of this approach with other baselines.

5.1 Training and Implementation

We compiled a huge corpus meta-data of articles on arXiv using [3]. We use this corpus to build labelled datasets for training of the two deep learning models in our pipeline - namely, BiLSTM CRF tagger and Siamese LSTM network.

We train the BiLSTM CRF tagger using an artificially created dataset consisting of citations and the corresponding labels or tags. Each citation was formed by concatenating author names, journal reference and year of publication in the exact order (as this is the most common format used in physics community). However, in practice, there are a variety of different styles and formats to list a citation. To make the network handle a variety of different writing styles and formats, we augmented the data by randomly interchanging the order in which the fields occur in the citation and sometimes, even dropping a field. For example, if the original order is “author names - title - journal - year”, we train the network with “title - author names - journal - year” or “author names - journal - year” in addition to the ideal reference. The different types of patterns which we used are listed in the Table 5.1. Although this allows to make the model

	Format	Number of samples
Training Set	A+J+Y	45000
	T+A+J+Y	4500
	A+T+J+Y	11250
	A+J+Y+T	3130
Validation Set	A+J+Y	3329
	T+A+J+Y	333
	A+T+J+Y	832
	A+J+Y+T	231

Table 5.1: Data augmentation used for training the Bi-GRU CRF network

somewhat robust to typos and variations in listing the citations, we argue that it seems critical to allow the model to focus more on the generic format compared to other rarely occurring ones. So, we keep the percentage of these distortions small (between 20-40%) as evident in Table 5.1. While training the model, we have five different labels in the dataset - author name (A), title (T), journal reference (J), year of publication (Y) and other (O) for punctuation and connecting words in the text. We decided to keep punctuation and connecting words (such as '.', ' ,', ';' and 'and') as these are seemed to be quite important cues for estimating the sequence of labels.

We use pre-trained GloVe embeddings [41] of 300 dimensions and also fine-tune the word vectors during training. We use 50 gated recurrent unit (GRU) cells in the bidirectional RNN layer and a dropout of 0.2 after the GRU layer. We randomly initialize the transition matrix of the CRF layer. We use a batch size of 64 and train the model for 5 epochs. The hyper-parameters were chosen based on the performance on 20% of the data which was kept aside as validation set.

For Siamese LSTM network, we consider a supervised learning setting where each training example consists of a pair of meta-data and citation say (x_{meta} and x_{cite}) along with corresponding label y. We derive this dataset from the arXiv meta-data dump. For obtaining the meta-data attribute of each sample we simply concatenate title, author names, year and journal references while the citations are constructed in a similar way as for the Bi-GRU CRF tagger. The meta-data is formatted as shown in Figure 5.1 wherein different fields are separated by '#SEP#'. The training sample has the label y = 1 if both the meta-data and the citation belong to the same paper else it has the label y = 0. For training, we introduce one positive sample and one negative sample for each paper wherein the

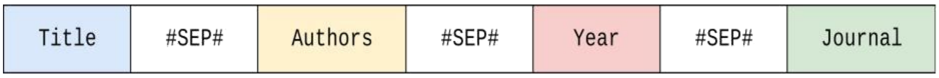


Figure 5.1: Format for meta-data input

negative example is obtained by swapping the citation with citation of some other paper. Though this leads to an exactly balanced dataset, it seems intuitive to include more negative models for training the network. So, we also consider more comprehensive ways to introduce more negative examples such as using citations wherein one of the author remains the same. The constructed data thus contains 193k samples. We keep 15% of the data for validation.

Since the complexity of the training data remains roughly the same, we use a similar set of hyper-parameters. But, we use a stacked bi-directional RNN with 50 LSTM units in the first layer and 50 GRU units in the second layer. This is followed by max pooling and a dropout of 0.15 before the final fully connected layer. We use a batch size of 256 and train the model until the performance on the validation set stops improving as shown in Figure 5.2. We use Adam optimizer with the default learning rate of 1e-3 for both the tasks.

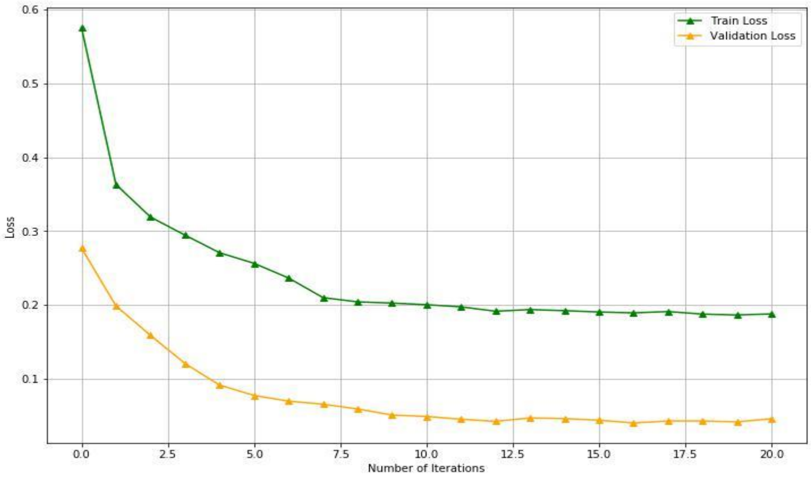


Figure 5.2: Training and validation loss vs number of iterations for the Siamese LSTM network

5.2 Performance Evaluation - KDD Cup 2003 Data Cleaning Task

In order to better understand the contribution of each component of the pipeline to the overall performance, we first look at the performance of each of the components individually. At the end, we would sum up these contributions and compare the results to the true citation graph released by KDD Cup 2003 for data cleaning task.

5.2.1 Regular Expression Matching for Extracting Citations

The citation extraction algorithm was able to extract around 960k references from 97.65% of the LaTeX sources using a compilation of several carefully crafted regular expressions. The expressions looked for the presence of common LaTeX tags such as "\begin{bibliography}" or words such as "References" to accurately find the bibliography section in the LaTeX file. After extracting the bibliography section, the algorithm was able to extract references from these sections by locating common identifiers used for listing references such as "\bibitem" as elaborated in Table 5.3. The percentage of files with which a particular pattern was found is enumerated in tables 5.2 and 5.3.

Regular Expression	No. of Occurrences
\begin{thebibliography} ... \end	28294
\begin{references} ... \end	5368
\ref ... .}\end	1209
Bibliography{ ... \end}	311
references} ... \bye	290
\begin{numberedbib ... \end	1

Table 5.2: Number of occurrences of regex patterns used for identifying the bibliography section

After extensive cleaning, we were able to extract around 960k citations from 34,640 LaTeX files. Unfortunately, sometimes a random part of text was extracted because it matched one of the patterns. For now, we keep such references and ignore this problem as it could be trivially solved by looking at the outputs of the tagger (as such texts probably won’t contain author names and year we can safely discard them).

Regular Expression	No. of Occurrences
\bibitem	31696
[NUMBER]	1310
\item	1016
\ref ... .}	865
\noindent	226
\bibit	16

Table 5.3: Number of occurrences of regex pattern used for extracting individual citations

5.2.2 Bi-GRU CRF Tagger

Table 5.4 shows the performance of the trained Bi-GRU CRF tagger on the constructed training and validation sets respectively. As it can be noted, the tagger achieves relatively high scores on the training and validation sets owing to the presence of clean, artificially created samples in the sets. Moreover, a high score on the validation set can be attributed to the fact that the training and validation sets are derived from the same parent data distribution.

The superiority of the BiGRU tagger becomes more clear when we investigate its performance on the raw references extracted from the hep-ph LaTeX sources. Unfortunately, due to the unavailability of true labels for these references, we cannot provide detailed empirical results. However, the perfect extraction of author names and year of publication with a high precision and low false positive rate is a key to the working of the pipeline as these entities are instrumental for matching citations to corresponding papers as shown in the next section. Thus, we find a way to approximately scrutinize the algorithm's performance for extracting authors and year. Checking if year is extracted correctly is equivalent to check if it's valid, which can be naively done using regular expressions. For extracted author names, we use a conservative approach and consider them to be correct they follow a certain format. Employing these tricks, Table 5.5 summarizes the approximate performance of the tagger

	Entity	Author	Year	Journal	Title	Other
F1	Training Set	99.09	99.99	98.67	98.59	99.74
Scores	Validation Set	99.10	99.99	98.70	98.55	99.77

Table 5.4: Performance of Bi-GRU CRF network on training and validation sets

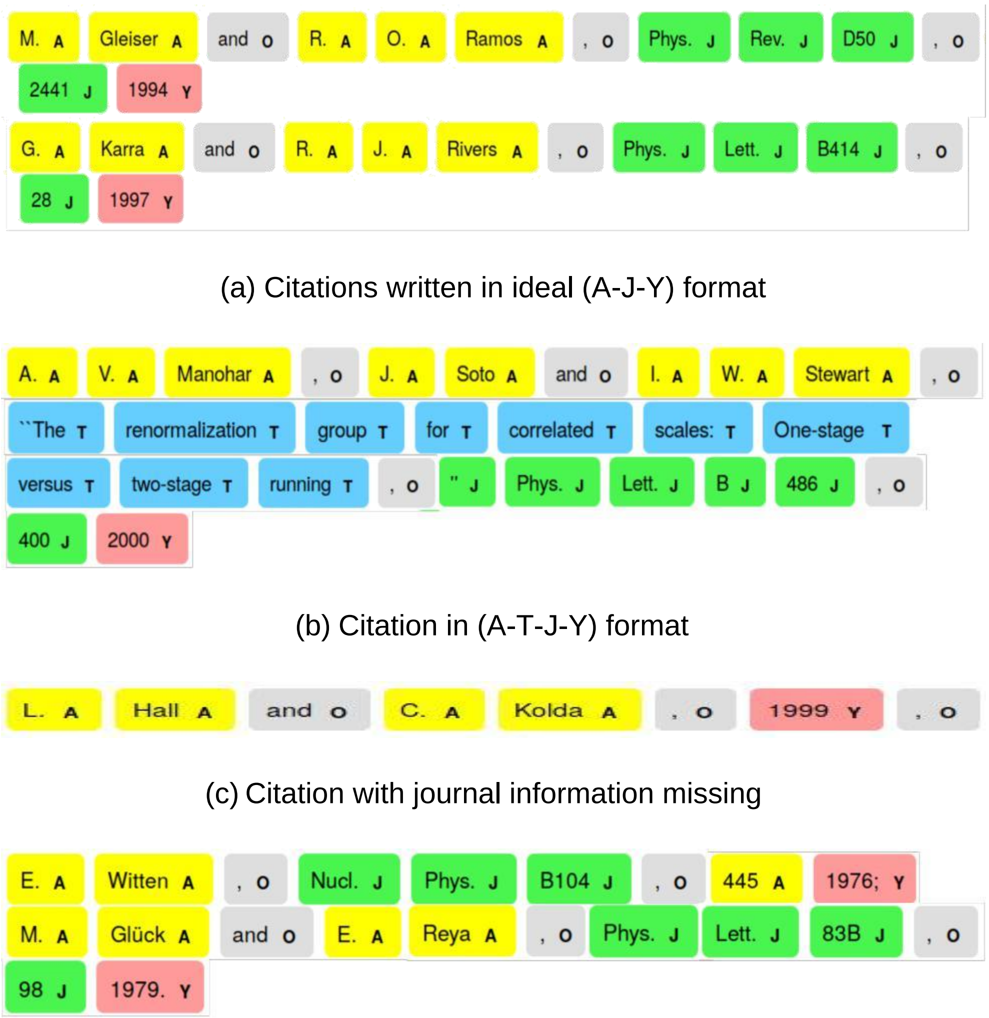
Entity	Precision
Author	96.5
Year	99.9

Table 5.5: Performance of Bi-GRU CRF network on hep-ph sources

for extracting author names and year. The slightly lower precision for author names is due to the presence of random texts mistakenly extracted as references by the first stage of the pipeline (for example, "References .....
] distribution are given by X (x) and . The actual choice of the interval width X is X not relevant since ..... range". We further employ a conservative final heuristic step on the identified author names to achieve a perfect precision.

A distinctive feature of the tagger is its robustness to missing information (Figure 5.3(c)) and different permutations of the entities in the citation. Figure 5.3(a) illustrates the tagged output for an example of citation in the format “Authors - Journal - Year” while Figure 5.3(b) shows the output for a “Authors - Title - Journal - Year” citation. Note that the presence of ‘O’ tag between different entities allows us to group multiple words corresponding to the same author name. Another significant advantage of the tagger is that it allows to separate multiple sources contained in the same citation as shown in Figure 5.3(d). Such cases usually occur in physics research papers, and can be solved simply by splitting at semicolon, if the next word after semicolon is an author name.

Nevertheless, a significant limitation of the model is the lack of variability in the constructed dataset used for training the tagger. Figure 5.4 shows the tagger output for a citation with some random text at the start. Due to the lack of such samples in the training set, the model tags "See for example,“ as author. However, we may note that there is no suitable tag for “See for example,” unless we want this to be labelled as ‘O’ (other). A more diverse and realistic training dataset would help surpass the performance of the tagger on such noisy counterexamples. We also considered employing BiGRU CRF CNN tagger [39] which utilizes character level features to handle out-of-vocabulary words, but it doesn’t lead to any major gains over BiGRU CRF tagger. Instead, we just use a separate token for out-of-vocabulary words. Thus, after the tagging step, we are left with 712k citations containing valid author name and year fields. Out of these 712k citations only 300k are found in arXiv meta-data after matching parsed author names and year.


M. A Gleiser A and O R. A O. A Ramos A , O Phys. J Rev. J D50 J , O
2441 J 1994 Y
G. A Karra A and O R. A J. A Rivers A , O Phys. J Lett. J B414 J , O
28 J 1997 Y
A. A V. A Manohar A , O J. A Soto A and O I. A W. A Stewart A , O
``The T renormalization T group T for T correlated T scales: T One-stage T
versus T two-stage T running T , O " J Phys. J Lett. J B J 486 J , O
400 J 2000 Y
L. A Hall A and O C. A Kolda A , O 1999 Y , O
E. A Witten A , O Nucl. J Phys. J B104 J , O 445 A 1976; Y
M. A Glück A and O E. A Reya A , O Phys. J Lett. J 83B J , O
98 J 1979. Y

(a) Citations written in ideal (A-J-Y) format

(b) Citation in (A-T-J-Y) format

(c) Citation with journal information missing

(d) Multiple citations (sources) on the same line

Figure 5.3: Output of Bi-GRU CRF network for various examples

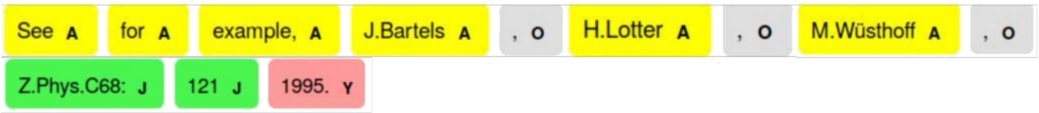


Figure 5.4: Citation for which Bi-GRU CRF network predicts wrongly

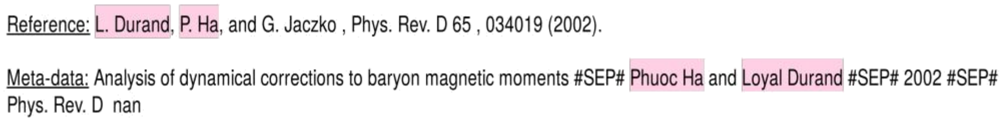
5.2.3 Siamese LSTM Network

With the way we formulated the task of matching citations to papers, the Siamese LSTM network needs to compare every citation with the meta-data of all the papers in the corpus. This is O(n^2) in terms of complexity and becomes infeasible for a large corpus such as arXiv. So, we divide the task of matching in two steps - first, filtering the list of meta-data of research papers using the extracted author names and then feed the reduced list along with citation to Siamese LSTM network for matching. This greatly reduces to amount of comparisons required to a number of papers of the same set of authors. We also consider filtering based on the extracted year. In case, where no paper is found, we also look for papers which are submitted a few years before (as this can be due to different versions of the paper).

Model	Accuracy
Siamese w. matched authors not removed	93.41
Siamese w. matched authors removed	**94.97**
Siamese w. multiple negatives	93.16
Siamese trained on hep-th meta-data	94.09

Table 5.6: Performance of different versions of Siamese LSTM network on hep-ph sources

With this approach, we achieve an accuracy of around 94% while matching citations to the paper. Table 5.6 summarizes the performance of different variations of this approach. We observe that author names are very dominant features for the Siamese network. In some examples, as shown in Figure 5.5, this drives the Siamese network to return a paper due to the presence of some dominant author feature in both the citation and the meta-data. In order to reduce such cases, given that we have already filtered the papers using author names, we remove author names from both citation and the meta-data before feeding them to Siamese network. This reduces the error rate to around 5%. Also, it can be seen from Table 5.6 that training with more negative examples doesn't benefit


Reference: L. Durand, P. Ha, and G. Jaczko , Phys. Rev. D 65 , 034019 (2002).

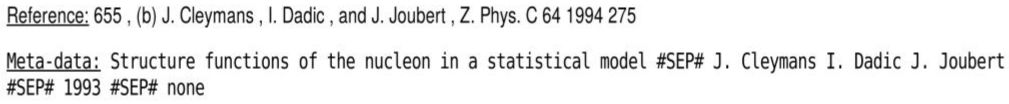
Meta-data: Analysis of dynamical corrections to baryon magnetic moments #SEP# Phuoc Ha and Loyal Durand #SEP# 2002 #SEP# Phys. Rev. D nan

Figure 5.5: Example of strong author features leading to mismatch

as compared to the best performing model which is trained with only one negative example for each paper.

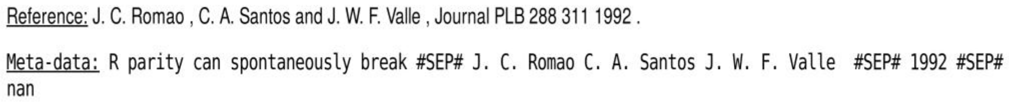
Looking at some specific examples, it can be clearly seen that the network performs well even in the cases with missing information, ambiguity as shown in Figure 5.6. Moreover, the network learns to create similar representations of different formats of the same citation.

After carefully examining the cases where the approach failed, we found a few cases in which the model was able to find mappings which were not present in the true citation graph constructed by SLAC. It is obvious that the result shown in Figure 5.7 is correct, but the corresponding edge is not present in the true graph and thus, negatively affects the computed prediction accuracy.


Reference: 655 , (b) J. Cleymans , I. Dadic , and J. Joubert , Z. Phys. C 64 1994 275

Meta-data: Structure functions of the nucleon in a statistical model #SEP# J. Cleymans I. Dadic J. Joubert #SEP# 1993 #SEP# none

Figure 5.6: Siamese LSTM predicts correctly even when journal name is missing


Reference: J. C. Romao , C. A. Santos and J. W. F. Valle , Journal PLB 288 311 1992 .

Meta-data: R parity can spontaneously break #SEP# J. C. Romao C. A. Santos J. W. F. Valle #SEP# 1992 #SEP# nan

Figure 5.7: Correctly predicted but not present in true citation graph

5.2.4 Overall Performance

The overall pipeline extracted around 300k in-corpus edges out of which 285k were correct, thus attaining a prediction accuracy of 94.97%.

Approach	No. of correctly predicted edges	Total no. of predicted edges	Accuracy	Symmetric difference size
#1-D. Vogel	-	-	-	421,582
#2-S. Sarawagi et. al.	40,600	175,800	23.09	516,242
#3-M. Cadot et. al.	-	144,087	-	538,013
Our pipeline	**285,064**	**300,099**	**94.97**	**151,549**

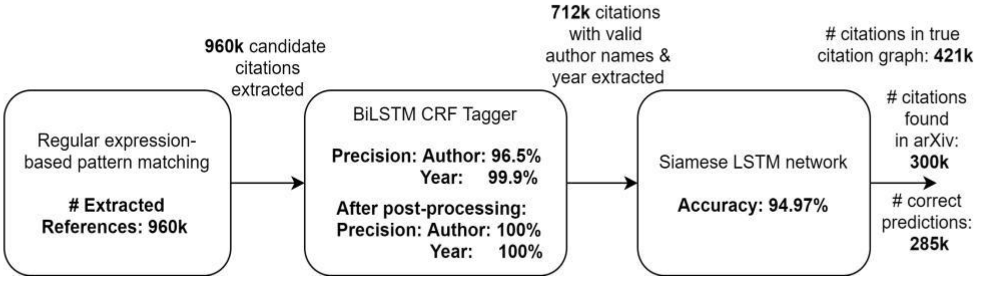
Table 5.7: Results on KDD Cup 2003 Data Cleaning task

Table 5.7 compares the performance of our pipeline with winning approaches [7] [8] in KDD Cup 2003. We observe an absolute boost of 71% in prediction accuracy and a substantial symmetric difference size improvement. Symmetric difference represents the disjunctive union between the sets of predicted edges and edges in target citation graph and is calculated as follows:

$$\text{Symmetric difference size} = j(E_{\text{true}} [E_{\text{pred}}) \text{ n } (E_{\text{true}} \backslash E_{\text{pred}})j \qquad (5.1)$$

where, E_{pred} is the set of edges extracted by the pipeline and E_{true} is the set of edges present in the true citation graph.

As evident from the Table 5.7, the pipeline predicted 285k edges correctly as opposed to 421k edges present in the true citation graph. This forms 67.7% of edges in the true graph. Thus, we have $jE_{\text{true}}j$ = 421k, $jE_{\text{true}} \backslash E_{\text{pred}}j$ = 285k and $jE_{\text{pred}}\ j$ = 300k. The failure to find the remaining 136k edges can be attributed to the existence of various sources of error in the pipeline. Firstly, the reference extraction step is not perfect and a major chunk of error is due to this step. Conversely, owing to the low false positive rate of the tagger, its contribution to the overall error should be minimal. Similar is the case with the Siamese network though it can be



712k citations with valid author names & year extracted
960k candidate citations extracted
# citations in true citation graph: 421k
Regular expression-based pattern matching
# Extracted References: 960k
BiLSTM CRF Tagger
Precision: Author: 96.5%
Year: 99.9%
After post-processing:
Precision: Author: 100%
Year: 100%
Siamese LSTM network
Accuracy: 94.97%
# citations found in arXiv: 300k
# correct predictions: 285k

Figure 5.8: Performance of individual stages of the pipeline

Entity	Precision
Author	96.1
Year	99.9

Table 5.8: Performance of Bi-GRU CRF network on hep-th sources

Model	Accuracy
Siamese trained on hep-ph meta-data	95.70
Siamese trained on hep-th meta-data	95.39

Table 5.9: Performance of Siamese LSTM network on hep-th sources

improved to achieve a perfect accuracy. Another source of error is the approach we use to compare our constructed citation graph with the true graph. The true KDD graph has random numbers as paper IDs. However, since the Siamese network maps matching citations to meta-data of papers obtained from arXiv, we get arXiv IDs at it's output. Thus, we translated these arXiv IDs to corresponding KDD IDs using the LaTeX sources for comparing the results with the true citation graph. We extracted abstracts and titles from the LaTeX sources to achieve a mapping between arXiv IDs and KDD IDs. But we were able to map only 33k papers out of 35k papers. This leads to situations where the pipeline correctly found an edge but since the mapping for that paper ID was either not present or was incorrect, it contributes to an incorrect edge (or a missing edge). Figure 5.8 consolidates the results obtained at different stages of the pipeline.

5.3 KDD Cup 2003 Citation Prediction Task

For the KDD Cup 2003 Citation Prediction task dataset, the citation extraction algorithm was able to find 827k references from 89% of the LaTeX sources. This is a significant drop as compared to 97% for hep-ph sources and as we will see, leads to a drop in the overall performance of the pipeline. Tables 5.8 and 5.9 encapsulate the results of BiLSTM CRF tagger and Siamese network (with matched authors removed) respectively on KDD Cup 2003 Citation Prediction task dataset. This demonstrates that the tagger and the Siamese network are able to generalise quite appreciably on hep-th articles without the need for re-training or fine-tuning with meta-data of hep-th articles. Furthermore, the performance achieved by Siamese network is competitive with that of model specifically trained with hep-th meta-data although training on a larger data would definitely be beneficial. Altogether, the pipeline correctly finds 230k citation edges between 29,555 papers with a prediction accuracy of 95%. This constitutes 66% of the edges in the true citation graph which contains 350k edges. This drop can be warranted to the lower performance while extracting references using regular expression matching.

Chapter 6

Conclusion and Future Work

Recent advances in deep learning for text have motivated the use deep networks for solving the problem of building a consistent bibliographic citation graph. In this work, we presented a generalized end-to-end 3-stage pipeline which exploits the benefits of deep learning for creating a consistent in-corpus citation graph of scientific articles on arXiv. The approach is able to tackle variations in writing styles and formats used for listing citations. On the whole, it extracts references and relevant information from the LaTeX source of a research paper with a not so state-of-the-art but still a commendable accuracy, thus solving the problem of reconstructing arXiv citation graph.

We believe that a lot of improvements can be made to the existing pipeline. The current reference extracting algorithm which is based on regular expression matching can be vastly improved. Possibly, one could experiment with the use of deep learning for the task of extracting references. Improving this extraction algorithm would surely lead to a major performance improvement since Bi-GRU CRF tagger and Siamese LSTM network are able to achieve noteworthy results. Training and evaluating the pipeline on a much larger corpus can also be looked at carefully. Specifically, [3] or [44] can be used to obtain a huge corpus of scientific articles not limited to a particular section of arXiv.

Bibliography

[1] J. Gehrke, P. Ginsparg, and J. Kleinberg, “Overview of the 2003 KDD Cup,” SIGKDD Explor. Newsl., vol. 5, no. 2, p. 149–151, Dec. 2003. [Online]. Available: https://doi.org/10.1145/980972.980992

[2] “KDD Cup 2003 - Data Cleaning Task.” [Online]. Available: http://www.cs.cornell.edu/projects/kddcup/data_cleaning_task.html

[3] C. B. Clement, M. Bierbaum, K. P. O’Keeffe, and A. A. Alemi, “On the Use of arXiv as a Dataset,” 2019.

[4] O. F. Guilarte, S. D. J. Barbosa, and S. Pesco, “Simplified Graph-based Visualization for Scientific Publication,” 2018.

[5] M. Eto, “Extended co-citation search: Graph-based document retrieval on a co-citation network containing citation context information,” Information Processing and Management, vol. 56, no. 6, p. 102046, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0306457318303637

[6] D. Jurgens, S. Kumar, R. Hoover, D. McFarland, and D. Jurafsky, “Measuring the Evolution of a Scientific Field through Citation Frames,” Transactions of the Association for Computational Linguistics, vol. 6, pp. 391–406, 2018. [Online]. Available: https://www.aclweb. org/anthology/Q18-1028

[7] S. Sarawagi, V. Vydiswaran, S. Srinivasan, and K. Bhudhia, “Resolving citations in a paper repository,” SIGKDD Explorations, vol. 5, pp. 156–157, 2003.

[8] M. Cadot and J. di Martino, “A Data Cleaning Solution by Perl Scripts for the KDD Cup 2003 Task 2,” SIGKDD Explor. Newsl., vol. 5, no. 2, p. 158–159, Dec. 2003. [Online]. Available: https://doi.org/10.1145/980972.980996

[9] G. Giuffrida, E. C. Shek, and J. Yang, “Knowledge-Based Metadata Extraction from PostScript Files,” in In Proceedings of the Fifth ACM Conference on Digital Libraries. ACM Press, 2000, pp. 77–84.

[10] A. Constantin, S. Pettifer, and A. Voronkov, “PDFX: Fully-Automated PDF-to-XML Conversion of Scientific Literature,” in Proceedings of the 2013 ACM Symposium on Document Engineering, ser. DocEng ’13. New York, NY, USA: Association for Computing Machinery, 2013, p. 177–180. [Online]. Available: https://doi.org/10.1145/2494266.2494271

[11] pdfextract - CrossRef Labs. [Online]. Available: https://www.crossref.org/labs/pdfextract/

[12] refextract - INSPIRE HEP. [Online]. Available: https://github.com/inspirehep/refextract

[13] Y. Ding, G. Chowdhury, and S. Foo, “Template Mining for the Extraction of Citation From Digital Documents,” in ASIAN DIGITAL LIBRARY CONFERENCE, 1999, pp. 47–62.

[14] I.-A. Huang, J.-M. Ho, H.-Y. Kao, and W.-C. Lin, “Extracting Citation Metadata from Online Publication Lists Using BLAST,” in Advances in Knowledge Discovery and Data Mining, H. Dai, R. Srikant, and C. Zhang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 539–548.

[15] Min-Yuh Day, Tzong-Han Tsai, Cheng-Lung Sung, Cheng-Wei Lee, Shih-Hung Wu, Chorng-Shyong Ong, and Wen-Lian Hsu, “A knowledge-based approach to citation extraction,” in IRI -2005 IEEE International Conference on Information Reuse and Integration, Conf, 2005., Aug 2005, pp. 50–55.

[16] M.-Y. Day, R. T.-H. Tsai, C.-L. Sung, C.-C. Hsieh, C.-W. Lee, S.-H. Wu, K.-P. Wu, C.-S. Ong, and W.-L. Hsu, “Reference metadata extraction using a hierarchical knowledge representation framework,” Decision Support Systems, vol. 43, no. 1, pp. 152 – 167, 2007, mobile Commerce: Strategies, Technologies, and Applications. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167923606001205

[17] C. Caragea, F. A. Bulgarov, A. Godea, and S. Das Gollapalli, “Citation-Enhanced Keyphrase Extraction from Research Papers: A Supervised Approach,” in Proceedings of the 2014 Conference on Empirical Methods

in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1435–1446. [Online]. Available: https://www.aclweb.org/anthology/D14-1150

[18] E. Hetzner, “A Simple Method for Citation Metadata Extraction Using Hidden Markov Models,” in Proceedings of the 8th ACM/IEEE-CS Joint Conference on Digital Libraries, ser. JCDL ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 280–284. [Online]. Available: https://doi.org/10.1145/1378889.1378937

[19] B. Ojokoh, M. Zhang, and J. Tang, “A Trigram Hidden Markov Model for Metadata Extraction from Heterogeneous References,” Inf. Sci., vol. 181, no. 9, p. 1538–1551, May 2011. [Online]. Available: https://doi.org/10.1016/j.ins.2011.01.014

[20] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” Proceedings of the IEEE, vol. 77, no. 2, pp. 257–286, Feb 1989.

[21] Hui Han, C. L. Giles, E. Manavoglu, Hongyuan Zha, Zhenyue Zhang, and E. A. Fox, “Automatic document metadata extraction using support vector machines,” in 2003 Joint Conference on Digital Libraries, 2003. Proceedings., May 2003, pp. 37–48.

[22] A. Kovacevic, D. Ivanovic, B. Milosavljevi´c, Z. Konjovic, and D. Surla, “Automatic extraction of metadata from scientific publications for CRIS systems,” Program: electronic library and information systems, vol. 45, pp. 376–396, 09 2011.

[23] J. Zou, D. X. Le, and G. R. Thoma, “Locating and parsing bibliographic references in HTML medical articles,” International Journal on Document Analysis and Recognition (IJDAR), vol. 13, pp. 107– 119, 2009.

[24] X. Zhang, J. Zou, D. X. Le, and G. R. Thoma, “A Structural SVM Approach for Reference Parsing,” in 2010 Ninth International Conference on Machine Learning and Applications, Dec 2010, pp. 479– 484.

[25] D. Tkaczyk, P. Szostek, M. Fedoryszak, P. J. Dendek, and Ł. Bolikowski, “CERMINE: automatic extraction of structured metadata from scientific literature,” International Journal on Document Analysis and Recognition (IJDAR), vol. 18, no. 4, pp. 317–335, Dec 2015. [Online]. Available: https://doi.org/10.1007/s10032-015-0249-8

[26] L. Romary and P. Lopez, “GROBID - Information Extraction from Scientific Publications,” ERCIM News, vol. 100, Jan. 2015. [Online]. Available: https://hal.inria.fr/hal-01673305

[27] P. Lopez, “GROBID: Combining Automatic Bibliographic Data Recognition and Term Extraction for Scholarship Publications,” in Research and Advanced Technology for Digital Libraries, M. Agosti, J. Borbinha, S. Kapidakis, C. Papatheodorou, and G. Tsakonas, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 473–474.

[28] J. D. Lafferty, A. McCallum, and F. C. Pereira, “Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data,” in ICML, 2001.

[29] I. Councill, C. L. Giles, and M.-Y. Kan, “ParsCit: an Open-source CRF Reference String Parsing Package,” in Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08). Marrakech, Morocco: European Language Resources Association (ELRA), May 2008. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2008/pdf/166_paper.pdf

[30] M.-Y. Kan, M.-T. Luong, and T. D. Nguyen, “Logical Structure Recovery in Scholarly Articles with Rich Document Features,” Int. J. Digit. Library Syst., vol. 1, no. 4, p. 1–23, Oct. 2010. [Online]. Available: https://doi.org/10.4018/jdls.2010100101

[31] A. Prasad, M. Kaur, and M.-Y. Kan, “Neural ParsCit: a deep learning-based reference string parser,” International Journal on Digital Libraries, vol. 19, 05 2018.

[32] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” Neural Comput., vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[33] E. Cortez, A. S. da Silva, M. A. Gonçalves, F. Mesquita, and E. S. de Moura, “FLUX-CIM: Flexible Unsupervised Extraction of Citation Metadata,” in Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, ser. JCDL ’07. New York, NY, USA: Association for Computing Machinery, 2007, p. 215–224. [Online]. Available: https://doi.org/10.1145/1255175.1255219

[34] J. Wu, K. Williams, H.-H. Chen, M. Khabsa, C. Caragea, A. Ororbia, D. Jordan, and C. Giles, “CiteSeerX: AI in a Digital Library Search

Engine," Proceedings of the National Conference on Artificial Intelligence, vol. 4, pp. 2930–2937, 01 2014.

[35] C. L. Giles, K. D. Bollacker, and S. Lawrence, "CiteSeer: An Automatic Citation Indexing System." ACM Press, 1998, pp. 89–98.

[36] W. Ammar, D. Groeneveld, C. Bhagavatula, I. Beltagy, M. Crawford, D. Downey, J. Dunkelberger, A. Elgohary, S. Feldman, V. Ha, R. Kinney, S. Kohlmeier, K. Lo, T. Murray, H.-H. Ooi, M. Peters, J. Power, S. Skjonsberg, L. Wang, C. Wilhelm, Z. Yuan, M. van Zuylen, and O. Etzioni, "Construction of the Literature Graph in Semantic Scholar," in Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers). New Orleans - Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 84–91. [Online]. Available: https://www.aclweb.org/anthology/N18-3011

[37] K. Lo, L. L. Wang, M. Neumann, R. Kinney, and D. S. Weld, "GORC: A large contextual citation graph of academic papers," 2019.

[38] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF Models for Sequence Tagging," 2015.

[39] X. Ma and E. Hovy, "End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF," 2016.

[40] J. P. C. Chiu and E. Nichols, "Named Entity Recognition with Bidirectional LSTM-CNNs," 2015.

[41] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://www.aclweb.org/anthology/D14-1162

[42] J. Mueller and A. Thyagarajan, "Siamese Recurrent Architectures for Learning Sentence Similarity," in Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, ser. AAAI'16. AAAI Press, 2016, p. 2786–2792.

[43] S. Chopra, R. Hadsell, and Y. Lecun, “Learning a similarity metric discriminatively, with application to face verification,” vol. 1, 07 2005, pp. 539– 546 vol. 1.

[44] D. R. Radev, P. Muthukrishnan, and V. Qazvinian, “The ACL Anthology Network Corpus,” in Proceedings of the 2009 Workshop on Text and Citation Analysis for Scholarly Digital Libraries, ser. NLPIR4DL ’09. USA: Association for Computational Linguistics, 2009, p. 54–61.