



WPI



BASIS TECHNOLOGY

Graduate Qualifying Project (DS 598)

Master of Science in Data Science Capstone

Entity Linking Prediction

Sponsors

Gil Irizarry
Vice President - Engineering
gil@basistech.com

Kfir Bar
Chief Scientist
kfir@basistech.com

Team Members







					
Vandana Anand vanand@wpi.edu 978-427-5212	Kratika Agrawal kagrawal@wpi.edu 774-701-9288	Jing Yu jyu5@wpi.edu 608-886-8578	Soumya Joshi sjoshi2@wpi.edu 508-615-8354	Min Huang mhuang3@wpi.edu 774-253-5410	Xinlu He xhe4@wpi.edu 774-253-8221

Table of Contents

Chapter 1: Introduction	4
1.1 Motivation and Background	4
1.2 Research Challenges and Gaps	4
1.3 Problem Statement	5
1.4 Project Objectives and Summary of Contributions	5
1.4.1 Project Objectives	5
1.4.2 Summary of Contributions	6
1.5 Project Plan	7
Chapter 2: Literature Review	8
Chapter 3: Methodology	9
3.1 End-to-End Entity Linking Model	9
3.1.1 Mention Detection	9
3.1.2 Candidate Selection	10
3.1.3 Global Score	10
3.1.4 Local Score	11
3.1.5 Final Score	11
3.2 Improving Entity Linking by Modeling Latent Relations between Mentions	11
3.2.1 Local Model	12
3.2.2 Global Model	12
3.3 DeepType: Multilingual Entity Linking by Neural Type System Evolution	12
3.3.1 Downloading and Transforming the Dataset	13
3.3.2 Building the Type System	13
3.3.3 Discrete Optimization of the Type System	13
3.3.4 Type Classifier	14
3.4 Robust Disambiguation of Named Entities in Text	14
3.4.1 Prior Probability	14
3.4.2 Mention-Entity Similarity	15
3.4.3 Entity-Entity Coherence	15
3.4.4 Graph Model and Algorithm	15
Chapter 4: Model Implementation	17
4.1 Dataset Overview	17
4.1.1 AIDA CoNLL-YAGO Dataset	17
4.1.2 Basis Technology Dataset	17

4.2 End-to-End Entity Linking Model	18
4.2.1 Workflow	18
4.2.2 Original and BT Dataset Preprocessing	19
Original Datasets	19
BT Dataset	20
4.2.3 Model Training and Testing	21
Training Model	21
Testing Model	21
4.2.4 Conclusion and Recommendation	22
4.3 Latent Relations Model	23
4.3.1 Workflow	23
4.3.2 BT Dataset Preprocessing	23
Original Dataset Preprocessing	24
Additional Dataset Preprocessing	24
4.3.3 Model Training and Testing	25
4.3.4 Conclusion and Recommendation	26
4.4 DeepType: Multilingual Entity Linking by Neural Type System Evolution	27
4.4.1 Downloading and Transforming the Dataset	27
4.4.2 Conclusion and Recommendation	27
4.5 Novel Approach with Robust Disambiguation	28
4.5.1 Generating Graph and Adjacency Matrices	29
4.5.2 Mention Entity Embeddings	29
4.5.3 Convolutional Neural Network	29
4.5.4 Conclusion and Recommendation	30
Chapter 5: References	31

Chapter 1: Introduction

1.1 Motivation and Background

Basis Technology (BT) is a company that aims to create a safer environment and more productiveness in the world by building Artificial Intelligence solutions for analyzing text, connecting data, and discovering digital evidence. For over twenty years, Basis Tech has provided solutions for businesses and the government to overcome some of the biggest challenges such as verifying identity, understanding customers, predicting global events, and even uncovering crime. Among the analytical software they developed include Rosette, a text analytics machine learning and deep neural networks hybrid to extract meaningful information from unstructured data (Basis Tech, 2020).

Rosette overall is an adaptable platform for text analytics & discovery. It offers various capabilities to efficiently analyze documents in more than 30 languages. Tasks supported by Rosette include chat translation, document categorization, parts of speech tagging, entity extraction, entity linking (EL), and more.

In the project affiliated with Worcester Polytechnic Institute (WPI) and BT called Graduate Qualifying Project (GQP), the focus will be on the EL aspect of Rosette. The Named-Entity Linking task aims to extract all named entities (ie. person, organization, location, etc) from a text document, called Entity Extraction/Entity Detection/Named-Entity Recognition, and link the identified mentions to an entity record in the WikiData database.

1.2 Research Challenges and Gaps

Unstructured data does not have an apt data model structure or a predefined organization method. They cannot be stored in a traditional database. Therefore, it is more difficult to analyze and not easily searchable. Data analytics tools such as Rosette aim to ease this problem and produce more valuable insights from the data. We will focus on improving Rosette's performance and conduct research on models that will provide an accurate prediction. There are many different models available so a challenge will be to pinpoint a model that will perform the best.

EL itself is a task that can be subdivided into Entity Extraction (EE) and Entity Disambiguation (ED). Thus, to link any entity to the corresponding record in the knowledge base, it should be ensured that the true entity in the knowledge base is chosen among all candidate entities in the knowledge base with the same name by following a procedure that takes several factors of the text into consideration.

Various research has been conducted in this area and uses different approaches to disambiguate the entity. Some consider the context of the entire text while others try to establish a relation between all entities mentioned in the text document. However, while every approach accomplishes a concept, it is still not perfect in the task of disambiguation.

1.3 Problem Statement

One important aspect of entity linking is to link the entity mentioned in the document to one and only one record of the WikiData. However, it poses a difficulty when more than one record in the WikiData database has the same name. For example: “Washington” in a text document can be referred to as either a US President George Washington or the US State Washington. Thus, an aspect of entity linking is to disambiguate the mention based on the entire text reference and then link it to a particular record in WikiData database.

Although BT has Rosette to efficiently perform entity detection and entity linking to the database, the goal is to improve and optimize the current procedure. This will be achievable by performing entity linking along with the utilization of entity disambiguation (ED) to achieve the best possible F1 or accuracy score. We would research and experiment with various machine learning or deep learning models and evaluate which one gives the best accuracy to recommend to BT. Then, we can investigate how to integrate this model with BT’s current procedure.

1.4 Project Objectives and Summary of Contributions

1.4.1 Project Objectives

Below are the project objectives that each student will learn and experience throughout the project. The students will come together as a team to produce and present the results after completion:

- Evaluation of a model including F1 scores, pros and cons, etc.
- Implementation and experiments on novel ideas based on each model
- Evaluation for improved models
- Summary of using neural networks in entity linking jobs
- Suggestions on future work

1.4.2 Summary of Contributions

Soumya Joshi	<ul style="list-style-type: none"> • Preprocessed the BT dataset • Trained the Latent model with different sizes of BT dataset • Tested the Latent model • Created Workflow Summary
Min Huang	<ul style="list-style-type: none"> • Preprocessed the BT dataset • Selected corresponding candidates for BT dataset • Calculated prior probabilities • Trained the Latent model with different sizes of BT dataset • Tested the Latent model
Vandana Anand	<ul style="list-style-type: none"> • Preprocessing datasets including AIDA • Trained and Tested the End-to-End Model • Compared scores and analyzed performance with other models • Designed Workflow Diagram
Xinlu He	<ul style="list-style-type: none"> • Preprocessed the BT dataset • Choose candidates for BT dataset • Trained the End-to-End Model with the AIDA dataset • Tested BT Data
Kratika Agrawal	<ul style="list-style-type: none"> • Tried to run the DeepType Repository • Imported Yago dataset to Postgres • Ran AIDA repository for generating graphs • Designed workflow diagrams • Built CNN model
Jing Yu	<ul style="list-style-type: none"> • Tried to run the DeepType Repository • Generated embeddings for mentions & entities • Built CNN model
All	<ul style="list-style-type: none"> • Wrote and edited the proposal, interim, and final paper • Worked on weekly presentation updates to BT and biweekly updates to the WPI GQP class

Table 1: Summary of Contributions

1.5 Project Plan

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
TASK	8-Sep-20	14-Sep-20	21-Sep-20	28-Sep-20	5-Oct-20	12-Oct-20	19-Oct-20	26-Oct-20	2-Nov-20	9-Nov-20	16-Nov-20	23-Nov-20	30-Nov-20	7-Dec-20
Team & Project Introduction														
Exploring tool Rosette														
Annotation Tool Review														
Read relevant papers														
Paper Presentations														
Compare Evaluation Metric														
Implementation: 3 Teams														
Transforming BT Dataset														
Evaluation on BT Dataset														
Discussion of Novel Approach														
Novel Approach Implementation														

Table 2: Project Timeline

Chapter 2: Literature Review

- End-to-End Neural Entity Linking, <https://arxiv.org/pdf/1808.07699.pdf>
 - Combine the Mention Detection (a.k.a Entity Linking) and Entity Disambiguation in one model
 - Consider both local score and global score, according to which to decide the final entity candidate
- DeepType: Multilingual Entity Linking by Neural Type System Evolution, <https://arxiv.org/pdf/1802.01021.pdf>
 - Builds the Type System for each Mention in the text document. Eg. Washington {Person, Politics, Place}
 - Classify to find the best Type System based on the reference in the text
 - Use the identified Type to disambiguate among Entities
- Latent Entity Modelling 1: <https://arxiv.org/pdf/1804.10637.pdf>
 - Apply representation algorithm to extract feature
 - Innovatively take into account latent relations between entities
- Latent Entity Modelling 2: <https://arxiv.org/pdf/2001.01447.pdf>
 - Utilizing the pretrained BERT model to produce similarity scores between entities and their mention context
- Repository to track the progress in Natural Language Processing (NLP), http://nlpprogress.com/english/entity_linking.html
 - Gain insight into the NLP procedures and understand best practices as well as how they are implemented since it is necessary for this project
- Robust Disambiguation: <https://www.aclweb.org/anthology/D11-1072/>
- Basis Technology Company Info: <https://www.basistech.com/about/>
 - Knowledge of what the company does and how our project will fit into their business needs to solve a challenge

We extracted our three baseline models from the above papers, and will go through their official implementation codes in Github to help us on our own implementation. Based on these materials, we can adjust these models and come up with new ideas to make a comparison or an improvement in entity linking.

Chapter 3: Methodology

Our overall project plan is to gather data and build a knowledge base, annotate the data, evaluate the current entity linking algorithms, design and build neural network models for entity linking as well as compare the performance of different neural network algorithms versus the current SOTA algorithms. There are two different approaches called End-to-End and Disambiguation-Only. End-to-End uses Named Entity Recognition and then disambiguates these extracted entities to the correct entry in a given knowledge base such as Wikidata, DBpedia, or YAGO. Disambiguation-Only directly takes gold standard named entities as input and only disambiguates them to the correct entry in a given knowledge base. Initially, we started with exploring three different models that includes End-to-End Entity Linking which utilizes the End-to-End approach, as well as DeepType and Entity Linking via Latent Relations which both use the Disambiguation-Only process. Later, with the knowledge gathered by these three models and also the Robust Disambiguation approach, we proceeded to work on a Novel Approach to perform the task of entity disambiguation.

3.1 End-to-End Entity Linking Model

End-to-End Entity Linking Model emphasizes the importance of the mutual dependency between Mention Disambiguation (MD) and Entity Disambiguation (ED) so that errors in one stage could be recovered by the next.

The research covers the following process:

1. Generate all possible spans that have at least one possible entity candidate.
2. Mention - Candidate pairs receive a context aware compatibility score based on word and entity embeddings coupled with a neural attention and global voting mechanisms.
3. During training, they enforce the scores of gold entities - mention pairs to be higher than all possible scores of incorrect candidates or invalid mentions, thus jointly taking the ED and MD decisions.

3.1.1 Mention Detection

Firstly, we used the following model to generate all the possible spans as the temperate mention for the future model. Here, we use bi-direction Long Short-Term Memory (LSTM) to capture the lexical information in character embeddings inside a word and to build context-aware word embeddings. After this, we obtain some possible mentions and can then select the entity candidates from it. Next, we will evaluate the selection using a score.

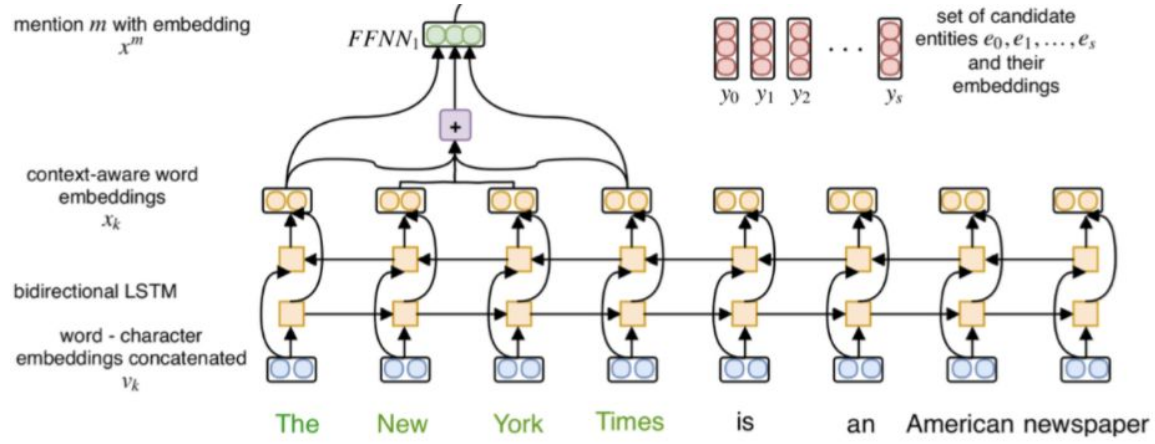


Figure 1: End-to-End model

3.1.2 Candidate Selection

Here, our goal is to select up to a number of entity candidates for each span, the possible mention. We select the candidates based on a conditional probability entity-map $p(e|m)$. Then we use fixed continuous entity representations, namely the pretrained entity embeddings.

3.1.3 Global Score

The global score focuses on the relationship among the entity candidates in the knowledge base and uses cosine similarity between the entity embeddings and the normalized average of all other voting entities' embeddings as the global score:

$$V_G^m = \{e | (m', e) \in V_G \wedge m' \neq m\}$$

$$y_G^m = \sum_{e \in V_G^m} y_e$$

$$G(e_j, m) = \cos(y_{e_j}, y_G^m)$$

Equation 1: Global scores

3.1.4 Local Score

The local score focuses on mentions and the relationship between mention and entity candidates. The local score is the output of a feed forward neural network (FFNN) with three separate inputs: the log prior probability of the mention-entity pair ($\log p(e_j | m)$), the LSTM similarity score of the pair, and the long range attention scores.

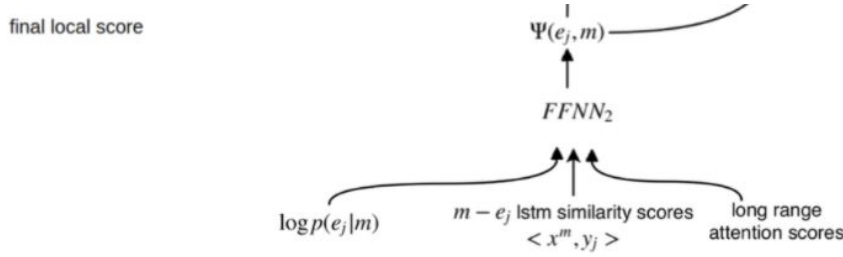


Figure 2: Calculating local score

3.1.5 Final Score

The final score consists of two parts, the local score and global score as well as a FFNN to get the final score.

$$\Psi(e_j, m) = FFNN_2([\log p(e_j | m); \langle x^m, y_j \rangle])$$

Equation 2: Calculating final score

3.2 Improving Entity Linking by Modeling Latent Relations between Mentions

The paper hypothesis states that relations used in Named Entity Linking do not need domain expertise. They can be recognized by using these relations as latent variables and optimizing the entity linking model. The paper uses Representation learning to learn mention embeddings, contexts and Named Entity linking relations. The paper uses the publicly available dataset AIDA-ConLL to train the model. To train a Latent Relations model, candidate selection is a major part of modelling relations as latent variables.

In the Latent Relations model, candidate selection is an important step as it helps select the perfect entity for the mention considering both the local context of the mention as well its relation to other mentions. The paper focuses on two major approaches, the local and global models.

3.2.1 Local Model

In the local model, only the local context of the mention is considered. Using this local context, a local score is calculated for each mention - entity pair and the pair with the maximum local score is selected for entity linking.

$$e_i^* = \arg \max_{e_i \in C_i} \Psi(e_i, c_i)$$

Equation 3: Local score Equation

Here, e_i is the entity i in the knowledge base and c_i is the local context of the mention.

3.2.2 Global Model

In the global model, both the local score and the weighted sum of all pairwise scores is considered. The pairwise scores are the scores between different mentions which relates to the relation between mentions. There can be k relations between mentions in a document.

$$E^* = \arg \max_{E \in C_1 \times \dots \times C_n} \sum_{i=1}^n \Psi(e_i, c_i) + \sum_{i \neq j} \Phi(e_i, e_j, D)$$

Equation 4: Global score equation

3.3 DeepType: Multilingual Entity Linking by Neural Type System Evolution

DeepType is a Disambiguation-Only type technique which explicitly integrates the symbolic information related to the identified mention by building a type system for them and choosing a type that best represents the entity corresponding to the text.

The DeepType system suggests a two-step solution to the problem:

1. Heuristic search for constructing a Type System
2. Fitting a classifier system to the type system

3.3.1 Downloading and Transforming the Dataset

The foremost task involved in the DeepType model is to download the entire Wikidata knowledge base and Wikipedia dumps in the English language. The Wikidata dataset is in JSON format with various details in several languages, while the Wikipedia/Wiki-Article dataset includes each document in XML format and is only in English.

Wikidata and Wiki-Article dumps are harnessed to meticulously extract various details from them including: anchor details, redirection links, category links, etc.

3.3.2 Building the Type System

A Type System is built using the knowledge graph or feature set that includes a group of type axes:

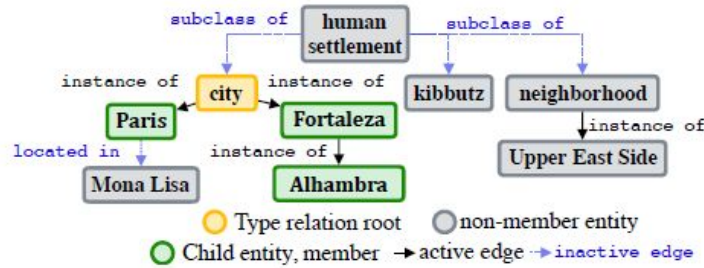


Figure 3: Type System building

3.3.3 Discrete Optimization of the Type System

A heuristic search or stochastic optimization is used to build the Type System, \mathcal{A} . A Learnability heuristic and an Oracle that quantify the disambiguation power of a proposed type system is used which is an estimate of how learnable the type axes in the selected solution will be.

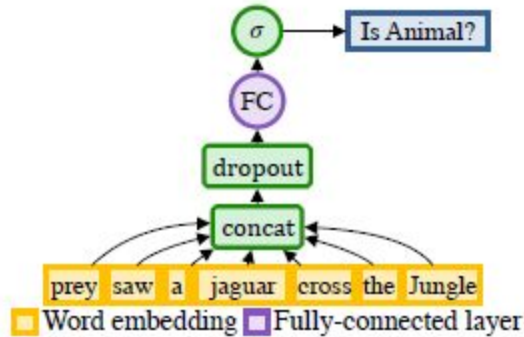


Figure 4: Type System Optimization

3.3.4 Type Classifier

Once discrete optimization has completed, a Type System is successfully built and is available. The goal for this classifier is to discover long-term dependencies in the input data that let it reliably predict types across many contexts and languages. For this reason, a bidirectional-LSTM with word, prefix, and suffix embeddings is selected to minimize the negative log likelihood of the per-token types for each type axis in the document D with L tokens.

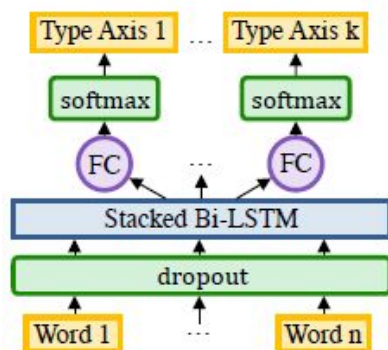


Figure 5: Type Classifier

3.4 Robust Disambiguation of Named Entities in Text

Robust Disambiguation is also a Disambiguation-Only type technique that unifies prior approaches into a comprehensive framework to combine three measures:

1. Prior Probability
2. Mention Context Similarity to Candidate Entity
3. Coherence among Candidate Entities of all Mentions

Using all these measures, a weighted graph is built and the ultimate goal is to compute a dense subgraph that approximates the best joint mention-entity mapping.

3.4.1 Prior Probability

Popularity of entities is seen as a probabilistic prior for mapping any mention name to an entity. Wikipedia provides frequencies for particular names in link anchor texts referring to specific entities using the number of inlinks. Anchor Text inlinks are considered to be the most effective measure for prior popularity which provides an estimate for the probability distribution over candidate entities.

For eg.: “Kashmir” refers to Kashmir (the region) in 90.91% of all occurrences and to Kashmir (the song) in 5.45% of all occurrences.

3.4.2 Mention-Entity Similarity

It is important to understand the context of the mention in order to link it to its name-based candidate entities. Thus, on the mention side, all tokens in the document are considered as the context for it and on the Entity side, knowledge base details are considered as its context and thus the similarity between mentions with all its candidate entities is calculated using either cosine similarity, KL divergence score, or Euclidean distance between the two.

Syntax based similarity is also considered to leverage the information about the immediate syntactic context in which a mention occurs. The model is trained on a large corpus to understand which type of words/verbs are more likely to co-occur with a mention and then ranks the candidate entities according to their compatibility with the verb.

3.4.3 Entity-Entity Coherence

Coherence among entities is a very important measure because most texts deal with a single or a few semantically related topics such as internet technology, rock music, politics or global warming, but not everything together.

Thus, using the semantic type system from the knowledge base like Yago, the coherence score among various entities can be calculated.

3.4.4 Graph Model and Algorithm

A graph is built using all the details from the text document and various scores calculated above:

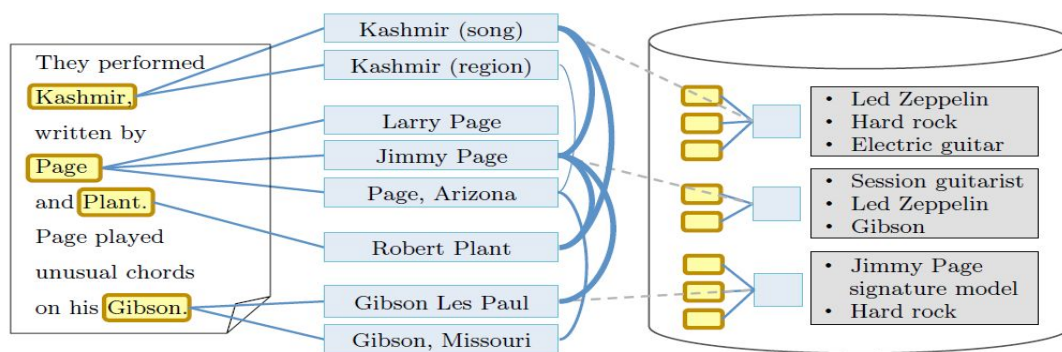


Figure 6: Graph model building

The graph has mention nodes and all candidate entity nodes. Mention and candidate entities nodes are connected with edges that are weighted links constituting combined scores of prior

probability and mention-entity similarity, while edges between various entities define the strength of the coherence among those entities.

A graph disambiguation algorithm is used to compute a dense subgraph such that the overall score for the subgraph is the highest among all possible subgraphs. This chosen subgraph contains exactly one entity for each mention, thus completing the disambiguation task.

Chapter 4: Model Implementation

4.1 Dataset Overview

4.1.1 AIDA CoNLL-YAGO Dataset

The AIDA CoNLL-YAGO dataset will be applied for training and testing the models while the BT dataset will be used for testing.

- This is a public dataset offered by the National Institute of Standards and Technology (NIST)
- This dataset is to be used during the implementation stage of the project to run various algorithms and compare its accuracies

4.1.2 Basis Technology Dataset

Basis Technology provides their own dataset including articles, mentions and gold entities information in a JSON format. Some additional information about the dataset include:

- It is internal data collected and prepared by BT
- It is already preprocessed following the preprocessing guidelines
- Annotations provided by BT using their in-house annotation tool
- Entity Extraction is followed as per the algorithm already built by BT
- Most of the entityIDs start with Q and indicate a link to the Wikipedia Knowledge Base

```
{
  "version": "1.1.0",
  "data": "On Wednesday, the total number of confirmed deaths linked to SARS-CoV-2 coronavirus infections surpassed 100,000 in the United States, Johns Hopkins University data indicated. The coronavirus causes COVID-19, a sometimes-fatal disease. The milestone came just under a month after the total number of confirmed infections in the United States surpassed one million on April 28.\n\nAs of Wednesday, the United States had the highest number of known infections, accounting for around 30% of world-wide coronavirus infections with over 1.6 million confirmed cases. The United States likewise had the highest number of confirmed deaths linked to the coronavirus, with the next highest country, the United Kingdom, reporting 37,542 deaths as of Wednesday.\n\nThe milestone came as states began to relax restrictions put in place during the COVID-19 pandemic. According to the University of Washington's Institute for Health Metrics and Evaluation, the coronavirus may cause roughly 32,000 more deaths in the United States by August 4.\n",
  "attributes": {
    "entities": {
      "type": "list",
      "itemType": "entities",
      "items": [
        {
          "mentions": [
            {
              "startOffset": 44,
              "endOffset": 50
            }
          ],
          "type": "SYMPTOM",
          "entityId": "Q4"
        }
      ]
    }
  }
}
```

Figure 7: Screenshot of the BT dataset

4.2 End-to-End Entity Linking Model

4.2.1 Workflow

The workflow for the End-to-End Model contains 3 steps: Training the model with the AIDA dataset, transforming the BT dataset to be compatible with the model, and finally testing the model with the BT dataset.

To train the model, we first needed to understand how the model works and the documentation on how to execute the model. We cloned the repository and first preprocessed the datasets including the AIDA dataset, to be compatible with the End-to-End model. Then we evaluated the F1 score to compare how well the model performed on each dataset.

The next task was to transform the BT dataset to be uniform with the trained datasets. In order to do this, we needed to generate all the possible mentions that have at least one possible entity candidate. In addition, each mention-candidate pair would receive a context-aware compatibility score based on word and entity embeddings. This involved needing to use the Word2Vec concept that involves converting all the words to numbers to be used in the testing phase. To test the model, the process would involve enforcing the scores of the gold entity-mention pairs to be higher than possible error scores, jointly taking into account the ED and MD decisions. Although we encountered some technical challenges with the BT dataset to use within the model, we were able to obtain good performance scores on the AIDA dataset.

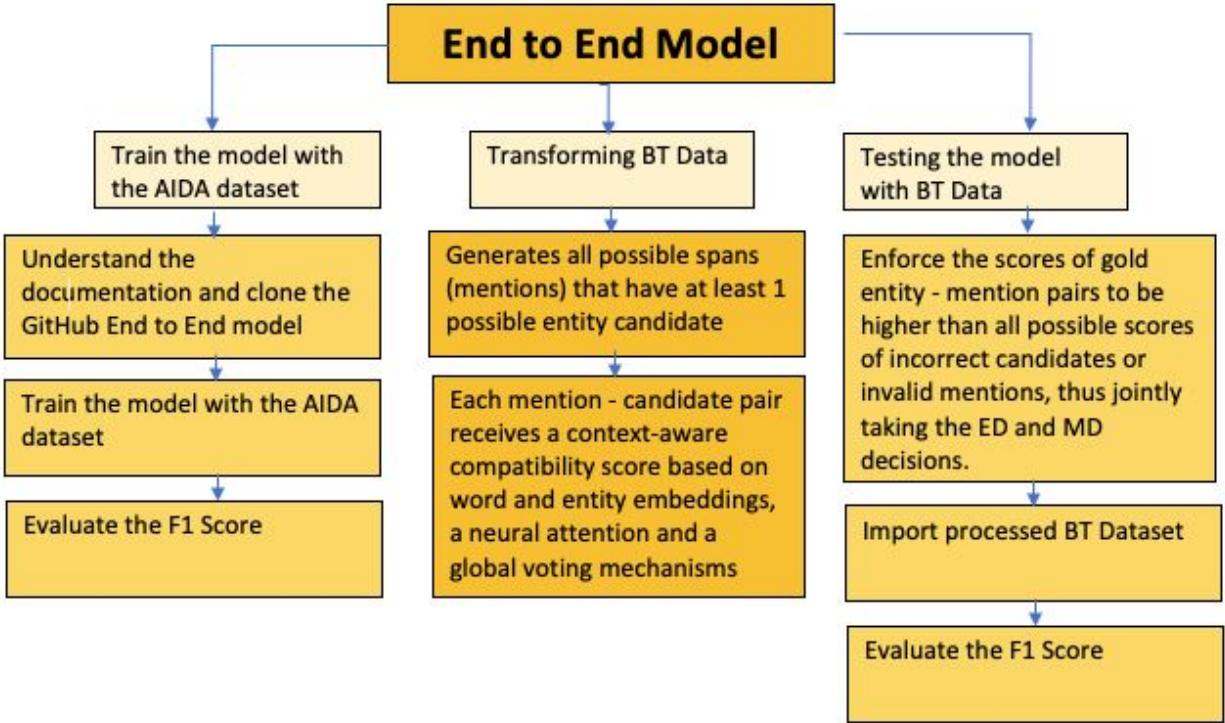


Figure 8: End-to-End Model Workflow Diagram

4.2.2 Original and BT Dataset Preprocessing

Original Datasets

In the End-to-End model, we utilized data from various sources: AIDA/CoNILL train, test, and validation datasets, MSNBC, Acquaint, Ace2004, Clueweb, and Wikipedia. We needed to transform this data from all of these sources into a uniform format in order to use them in the model and as training data.

The original format of the data from the above sources was a raw text document along with its XML file that contains the document name, mention name, the wikipedia article name linking to the mention, and the positions of the mention in the raw text as shown in Figures 9 and 10.

Text

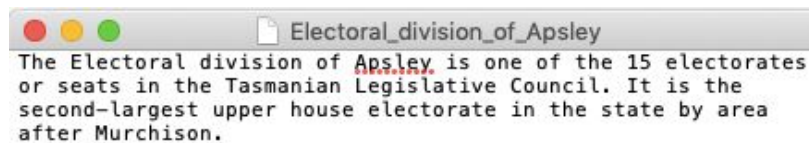


Figure 9: Raw text document for Wikipedia data

XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<wikipediaData.entityAnnotation>
  <document docName="Electoral division of Apsley">
    <annotation>
      <mention>Tasmanian Legislative Council</mention>
      <wikiName>Tasmanian Legislative Council</wikiName>
      <offset>78</offset>
      <length>29</length>
    </annotation>
    <annotation>
      <mention>Murchison</mention>
      <wikiName>Electoral division of Murchison</wikiName>
      <offset>184</offset>
      <length>9</length>
    </annotation>
  </document>
```

Figure 10: XML file for Wikipedia data

After preprocessing the data, the final format transformed into containing:

A start mark for every document with its title: DOCSTART_fileName

Every word, punctuation, or special character in the raw text in its own row

Every mention in its own row marked with the keyword “MMSTART_Wikiid” in the row before it and “MMEND” after it

The *NL* mark in its own row between every paragraph in the raw text document

The AIDA dataset contained these features in addition to the URL linking to the wikipedia article.

BT Dataset

To test the BT data, we needed to transform it to be consistent with the datasets we already processed and compatible with the End-to-End model. An added challenge was that the BT dataset was in a JSON format. We reworked the preprocessing logic so we could obtain a similar output as the training datasets as explained above.

For BT’s dataset, there are two kinds of ID’s used for linking purposes. One is the Wiki ID and the other is the ID in BT’s own database. We deleted the links with ID’s from BT’s database so that we could focus on using the wiki id for this project’s purpose.

As a result, we processed 70 documents provided by BT. There are a total of 3,114 mention-entity pairs and deleted 207 pairs containing ID’s from the BT database.

4.2.3 Model Training and Testing

Training Model

After training the End-to-End model we obtained the results shown in Figure 11. Overall, ED performed a lot better than EL. The training model for different datasets using both ED and EL performed a bit worse than the Latent Relations model, whose lowest performing dataset was Wikipedia with an accuracy of 77%. However, the ED model itself shows a lower score of around 74-75% while the EL has a much lower score at around 41-43%.

Entity Linking Training

```
Evaluating EL datasets
Best validation threshold = 0.053 with F1=90.1
aida_dev
micro P: 90.1  R: 90.2      F1: 90.1
macro P: 88.1  R: 88.3      F1: 88.2
aida_test
micro P: 83.6  R: 82.1      F1: 82.8
macro P: 83.8  R: 84.3      F1: 84.1
aida_train
micro P: 96.0  R: 95.4      F1: 95.7
macro P: 95.3  R: 95.2      F1: 95.3
ace2004
micro P: 19.3  R: 69.0      F1: 30.2
macro P: 21.3  R: 61.4      F1: 31.6
aquaint
micro P: 38.2  R: 43.3      F1: 40.6
macro P: 40.1  R: 41.8      F1: 40.9
clueweb
micro P: 45.7  R: 49.1      F1: 47.3
macro P: 53.6  R: 49.0      F1: 51.2
msnbc
micro P: 78.1  R: 76.3      F1: 77.2
macro P: 79.5  R: 74.5      F1: 76.9
wikipedia
micro P: 40.9  R: 42.8      F1: 41.8
macro P: 44.1  R: 43.4      F1: 43.7
((end2end_neural_el_env) [wpi-project@wpi-gcp-2021-2
```

Entity Disambiguation Training

```
Evaluating ED datasets
Best validation threshold = -0.037 with F1=93.8
aida_dev
micro P: 94.5  R: 93.1      F1: 93.8
macro P: 93.1  R: 91.9      F1: 92.5
aida_test
micro P: 89.2  R: 85.4      F1: 87.2
macro P: 90.0  R: 88.1      F1: 89.1
aida_train
micro P: 97.3  R: 96.1      F1: 96.7
macro P: 97.0  R: 95.9      F1: 96.5
ace2004
micro P: 92.6  R: 83.9      F1: 88.1
macro P: 93.8  R: 85.1      F1: 89.2
aquaint
micro P: 92.4  R: 87.2      F1: 89.7
macro P: 92.4  R: 86.7      F1: 89.4
clueweb
micro P: 83.2  R: 72.3      F1: 77.3
macro P: 82.8  R: 72.6      F1: 77.4
msnbc
micro P: 94.4  R: 90.3      F1: 92.3
macro P: 95.4  R: 91.1      F1: 93.2
wikipedia
micro P: 78.2  R: 70.9      F1: 74.4
macro P: 78.7  R: 72.2      F1: 75.3
((end2end_neural_el_env) [wpi-project@wpi-gcp-2021-2
```

Figure 11: End-to-End model training and testing

Testing Model

After testing the base model using the AIDA test dataset, we obtained the scores for ED and EL shown in Tables 3 and 4. Similar to the training model performance, ED once again performed better than EL. The EL scores are very similar to the Latent Relations testing score with around 55% accuracy. However, these scores are significantly improved when using all the spans of the text document.

Entity Disambiguation	Train Score	Test Score	Test Precision	Test Recall
Best Scores	92.2%	85.7%	88.5%	83%

Table 3: ED Testing

Entity Linking	Train Score	Test Score	Test Precision	Test Recall
Best Scores	62.9%	56.8%	57.5%	56.1%
Using all Spans Best Score	89.1%	80.1%	83.5%	76.9%

Table 4: EL Testing

4.2.4 Conclusion and Recommendation

The future scope of this project is to test the BT dataset with the End-to-End model and assess the accuracy since the dataset cannot be inserted into the model directly. The reason is that we cannot take full advantage of the End-to-End model if we already use the Mention Extractor because we could no longer use the Mention Detection part of the model, which could cause issues. As a solution and future work, we could rearrange the implementation model to be compatible with BT's data structure.

The End-to-End model gave promising accuracy scores using the AIDA dataset as it contains more than 500K mentions. Since there is a lower amount of BT data, thus a lower amount of mentions at around 3,000, we expect its testing accuracy score to be about the same or lower than AIDA. If the BT data shows reasonable results, we could run the model on BT's server and integrate with existing processes to improve the company's entity procedures.

4.3 Latent Relations Model

4.3.1 Workflow

Our model implementation workflow includes four major activities. The first activity is the Github repository cloning and implementation which consists of the training model mentioned in the paper. The second activity consists of calculating the prior probability, which is a measure of the probability a mention may belong to the particular entity. In our model, the prior probability is an input to the neural network we have created for the training. Since the original code is in Lua, we have to code it from scratch in python. The third activity includes tasks related to testing the BT dataset with the trained model. The fourth activity is to increase model performance by including a chunk of the BT dataset in the training process so that the micro F1 accuracy score can improve.

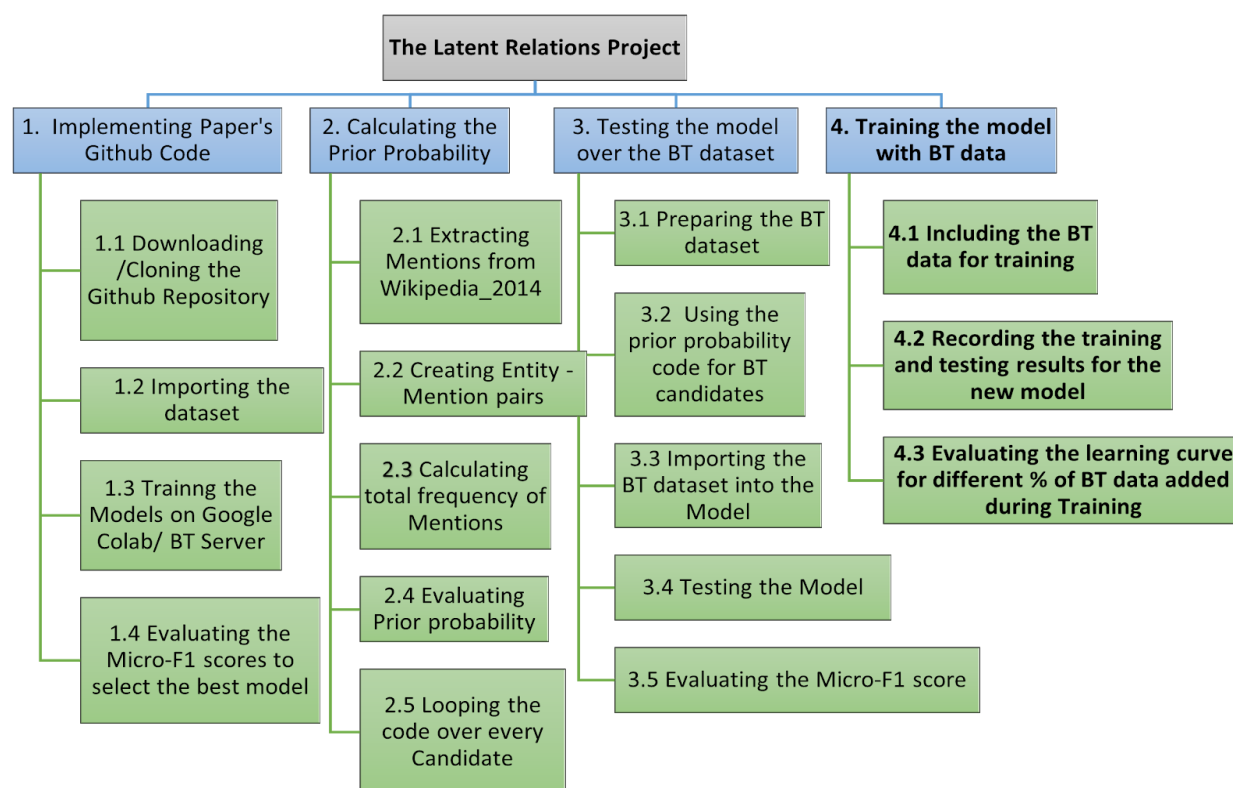


Figure 12: Latent Relations Model Workflow Diagram

4.3.2 BT Dataset Preprocessing

For the Latent Relations model, the input information should include the Mention, Context, Candidates, Gold and Secondary Context. In order to extract and collect these data, we conducted a two-phase data processing procedure. In phase 1, we did basic processing work on

the original BT dataset. In phase 2, we applied an additional dataset and then processed this dataset to get Candidate information.

Original Dataset Preprocessing

Based on the original dataset, we simply extracted mentions from the articles using their position indexes. For the entities, we only kept the entity_id starting with Q and applied the wbgetentities API to get the entity name. For the context information, we extracted both the right context and left context of each mention, cleaned the context by removing any punctuations, and limited the maximum words to 100. To get the secondary context information, we split each sentence into a words list and then used a dictionary to store all the sentences. Moreover, we used the list.index() function to get the position of each mention and another dictionary to store all of the mentions.

Additional Dataset Preprocessing

In order to select Candidates for each mention, we needed to apply an additional dataset that contains all the articles in the Wikipedia Knowledge Base from 2014. These articles are anchored and saved as a text file. By recognizing the element `<a>` and the href attribute, we could extract corresponding entities for each mention. For the sample in Figure 13, there are two mention-entity pairs which are (Irish republican, Irish republicanism) and (Derry, Derry).

```
e.g. Séamus O'Doherty (11 June 1882 - 23 August 1945) was an <a href="Irish republicanism">Irish republican</a>.\n\nSéamus O'Doherty was born on 11 June 1882 in <a href="Derry">Derry</a>.
```

Figure 13: A Sample of textWithAnchorsFromAllWikipedia2014Feb.txt

Apart from the entities, we still needed their prior probabilities for our model. For each mention-entity pair, the prior probability equals its frequency dividing the frequency of all mention-entity pairs for the same mention, marked as $P(e|m)$. To get entities and their prior probabilities, we first extracted the mention-entity pairs and recorded their frequency at the same time. Then we stored them into the dictionary, calculated the total frequency, and saved it as the frequency.json file. Lastly, we integrated the BT dataset with this frequency dataset by selecting the corresponding entities and calculating the prior probability for each mention-entity pair.

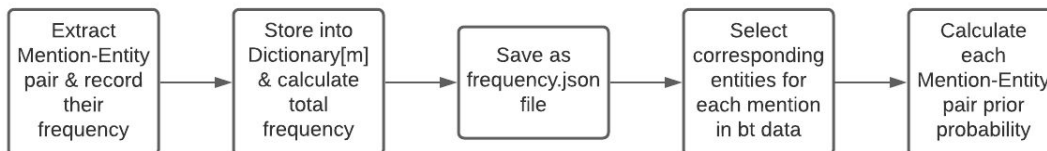


Figure 14: Workflow of candidates generation

4.3.3 Model Training and Testing

In the original paper, the training dataset is the AIDA-CoNLL dataset, which contains 953 documents. For the BT dataset, we were gradually able to obtain a total of 70 documents. Among the 70 documents, we used 60 documents for training and 10 documents for testing. We designed several training procedures by adding 10 documents each time we ran the model. For each training process, we ran 3 rounds and recorded the testing results as well as the average values, shown in Table 5.

Training docs	Original	Original+ 10docs	Original+ 20docs	Original+ 30docs	Original+ 40docs	Original+ 50docs	Original+ 60docs
Testing1 Results	0.444	0.426	0.479	0.489	0.546	0.482	0.584
Testing 2 Results	0.440	0.507	0.482	0.461	0.553	0.542	0.528
Testing 3 Results	0.465	0.454	0.419	0.486	0.465	0.493	0.542
Avg Testing Results	0.449	0.462	0.460	0.479	0.521	0.506	0.551

Table 5: Testing results

We drew a box-and-whisker diagram as well as a line chart for showcasing the testing results. For each document combination, both the interval and average value are shown in Table 5. The overall F1-score is not high, which could be due to two main problems. One of the problems is that the BT dataset is new, therefore, the context of each mention is new. However, we generated candidates based on an old version of Wikipedia Knowledge Base in 2014, making the context information less effective. Another problem is that the BT dataset is too small. Although we added the BT dataset to the training data, the main training data is the AIDA-CoNLL dataset that is much older. This inconsistency between the two datasets could lead to a lower score. However, the learning curve in Figure 15 still shows that the Latent model might be a great choice for the BT dataset. Although the interval for each document combination is large, the learning curve shows an increasing trend for the F1-score with a larger BT dataset. This is a positive signal and displays that the model has the potential to increase the accuracy of BT's current process.

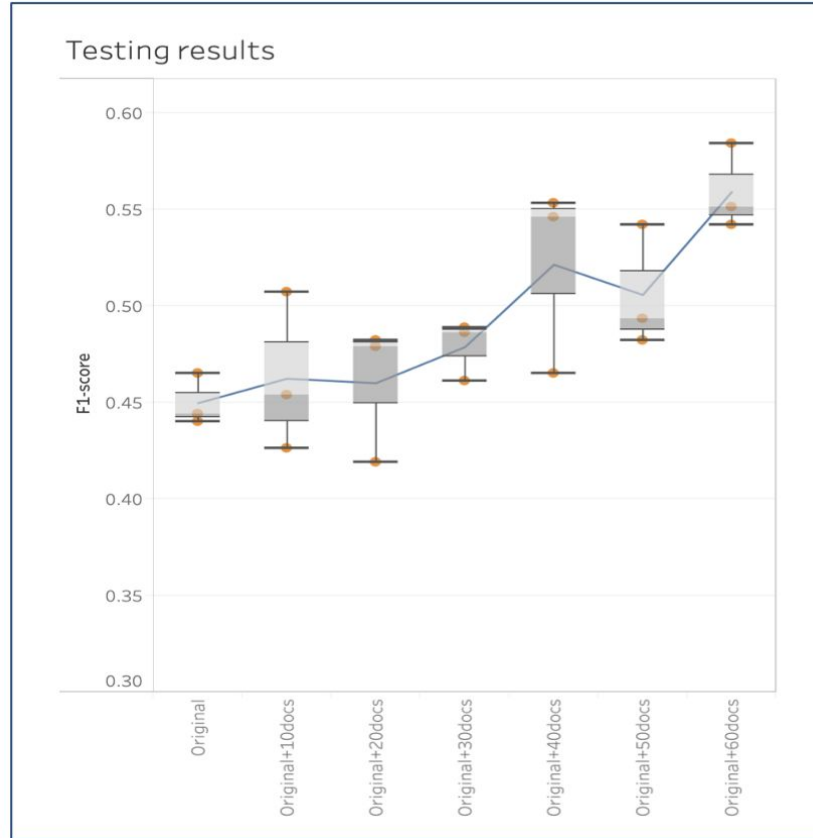


Figure 15: Learning curve

4.3.4 Conclusion and Recommendation

Based on our research, we successfully applied the BT dataset to the Latent Relations model. The results also indicate better performance with larger BT datasets. In the future, a larger BT dataset is required for further improvement. If the BT dataset is large enough, we could generate candidates based on the latest version of Wikipedia Knowledge Base and train the model only on BT dataset. In this way, the contexts and the candidates would be more relevant to the BT dataset. Overall, the Latent model would be a great choice to improve upon BT's current procedures.

4.4 DeepType: Multilingual Entity Linking by Neural Type System Evolution

4.4.1 Downloading and Transforming the Dataset

The DeepType model implementation requires the latest Wikidata and Wiki-Article dumps to be downloaded and extracted so that various information like anchor links, category links, redirection links can be harnessed to build a huge Knowledge Graph.

We have been able to perform the following tasks:

1. Downloaded and Extracted the Wikidata entities dataset. The dataset is in JSON format with various details in several languages.
2. Downloaded and Extracted all Wiki-Articles in the English language. The dataset is in XML format.
3. Explored Wikidata entities and fetched only the English language details while discarding the others.
4. Learned about the PID that is used in wikidata to define various properties of entities.

4.4.2 Conclusion and Recommendation

The task of processing various extracted dumps for building the project graph has been an issue as wiki dumps are huge in size and need a tremendous amount of memory in the system. This became a problem for us with limited resources.

We also tried to explore the dataset on the server BT provided and the WPI turing server, but those resources were also not large enough to accommodate the data processing.

As suggested by the DeepType model developers in their github repository, which is already 2 years old, implementation of the model needs a minimum of 1 terabyte of space on the local machine. However, in the past 2 years, the data has grown to become much larger in size and now needs a minimum of 1.5 terabytes of space. Thus, it was difficult to process the wiki dumps that are needed to generate knowledge graphs for further processing.

Thus, we suggest that if a parallel processing cluster is set up that can use a standard map reduce technique to harness required details from wiki dumps simultaneously on multiple machines using shared resources, then it is possible to process such a large dataset to generate knowledge graphs and consequently, proceed with the disambiguation process.

4.5 Novel Approach with Robust Disambiguation

After learning various concepts about models in other papers, we worked to come up with a new approach for the disambiguation task called Robust Disambiguation.

The implementation overview of this approach is shown:

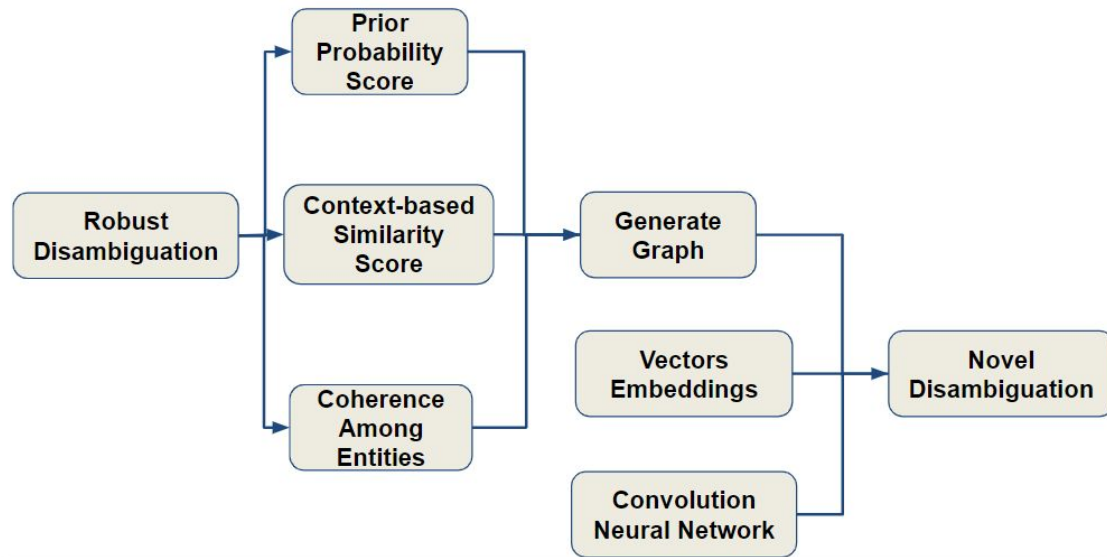


Figure 16: Workflow of the Novel Approach

The model is an extension of the Robust Disambiguation technique, which also takes mentions and entities' embeddings into consideration. All these features are passed on to a Convolutional Neural Network (CNN), a deep learning technique traditionally used to learn features of an image, graphs, and embedding features to perform the disambiguation task.

The entire development cycle and major tasks involved can be seen as:

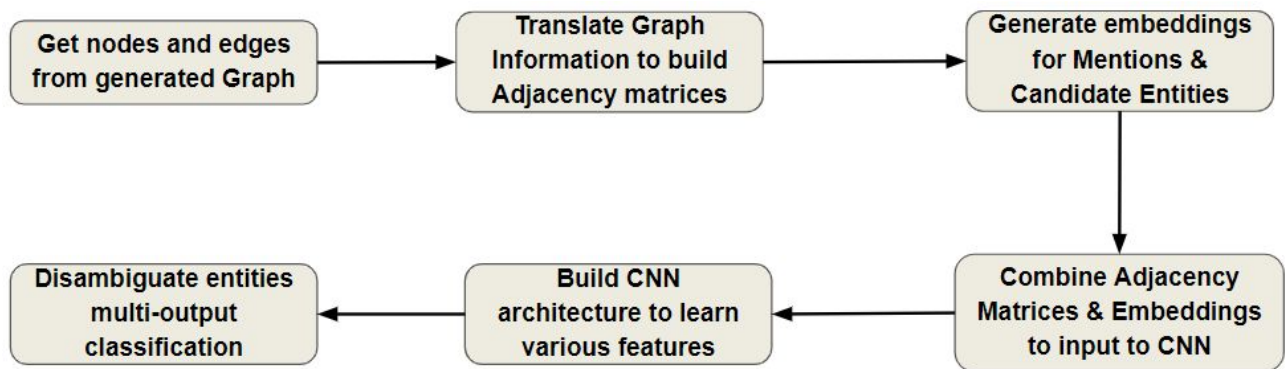


Figure 17: Entire Development Cycle and major tasks

4.5.1 Generating Graph and Adjacency Matrices

Robust Disambiguation takes various measures into consideration in order to perform disambiguation, such as prior probability, context-based similarity, and coherence among different entities. A graph is then generated from utilizing all of this information.

Setup for the Robust Disambiguation model can be done using details in the Github repository code for the AIDA dataset. The graph can be extracted for various documents from the same repository using the class `GenerateGraph.java` in the package called `mpi.aida.graph`.

First, `gdata` should be obtained from the function `run()` in the class, which consists of all details in the graph. The generated graph contains all the information about mention nodes, candidate entity nodes, and connecting links.

Next, set the *entitiesPerMentionConstraint* flag to be true and keep the constraint candidate entities per mention at 5 to build the graph (keeps the top 5 candidate entities for each mention based on the prior probability score and the mention context similarity score).

The graph can be generated by running the following command in the terminal:

```
java -Xmx4G -cp aida.jar mpi.aida.CommandLineDisambiguator GRAPH <INPUT-FILE>
```

Now, for our new approach, we need to translate the generated graph to adjacency matrices. Mention to Entity links can be converted to Mention-Entity weighted adjacency matrices and Entity to Entity graph links can be converted to Entity-Entity weighted adjacency matrices for each document.

4.5.2 Mention Entity Embeddings

Wikipedia2Vec embeddings are used to generate embeddings for various identified mentions and their candidate wikipedia entities. The Wikipedia2Vec pretrained model embeds mentions using the Word2Vec embedding and all candidate entities using the Wikipedia2Vec embeddings of 100 features.

4.5.3 Convolutional Neural Network

The generated adjacency matrices and embedding matrices for each document are combined in a systematic fashion to be passed as input to the CNN.

CNN is a deep network with 3 convolutional layers followed by tanh activation and dropouts.

When the network converges and becomes flattened, we concatenate it with each mentions' and related candidate entities' embeddings to disambiguate it.

The model is expected to train on a very large dataset.

4.5.4 Conclusion and Recommendation

The graphs can be generated for various AIDA-CoNLL data documents and can be translated to adjacency matrices. Embeddings can also be generated for them and passed into the CNN model for training.

We believe that when the CNN model is trained on a large training set and the shared weights will be learned, CNN will provide a very good F1 score on the test dataset.

Because of its unique nature of learning ways to disambiguate using weight sharing, it might perform even better on the BT dataset without specific training on the BT data.

Chapter 5: References

- [1] End-to-End Neural Entity Linking, <https://arxiv.org/pdf/1808.07699.pdf>
- [2] DeepType: Multilingual Entity Linking by Neural Type System Evolution, <https://arxiv.org/pdf/1802.01021.pdf>
- [3] Latent Entity Modelling 1: <https://arxiv.org/pdf/1804.10637.pdf>
- [4] Latent Entity Modelling 2: <https://arxiv.org/pdf/2001.01447.pdf>
- [5] Repository to track the progress in Natural Language Processing (NLP), http://nlpprogress.com/english/entity_linking.html
- [6] End-to-End Github Link: https://github.com/dalab/end2end_neural_el
- [7] Deeptype Github Link: <https://github.com/openai/deeptype>
- [8] Latent Github Link: <https://github.com/izuna385/Entity-Linking-Recent-Trends>
- [9] Basis Technology Company Info: <https://www.basistech.com/about/>
- [10] Training Entity Embeddings: <https://github.com/dalab/deep-ed/>
- [11] Latest Wikidata dumps: <https://dumps.wikimedia.org/wikidatawiki/entities/latest-all.json.bz2>
- [12] Latest English language Wikipedia dumps: <https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2>
- [13] Robust Disambiguation: <https://www.aclweb.org/anthology/D11-1072/>
- [14] Robust Disambiguation Github code: <https://github.com/codepie/aida>
- [15] Wikipedia2Vec Embedding: <https://wikipedia2vec.github.io/wikipedia2vec/>