Identifying sentences that convey meaningful semantic relation between entities within the sentence

Detailed report

Maor Hornstein, 300278520

Submitted as a report for the Advanced Project Workshop course (89985), BIU, 2023

The purpose of this file is to document the technical decisions made during the course of the project, along with the corresponding findings and conclusions. It provides guidance for future efforts to anyone who will continue working on the project, if found relevant.

The report is structured into 10 chapters that follow a chronological order, wherein each chapter addresses a different step of working with the data.

When code is used within any of the chapters, the corresponding reference from the project's git repository will be provided. The outputs will be shared through a link to the relevant directory in the project's storage drive.

Table of Contents

1. Choosing the Dataset.......... 3
2. Data Cleaning.......... 4
3. Dataset Creation..........5
3a. Data Processing and Filtering.......... 5
3b. Encoding Creation..........10
4. Exploratory Data Analysis (EDA)..........11
5. Model 1 - Classification on top of BERT.......... 11
6. Model 2 - Classification on top of BERT with labels..........11
7. Baseline models..........12
8. Classification Models Performances - Conclusions..........13
8a - Models comparison..........13
8b Discovering the Optimal Configuration for Network Efficiency..........13
8c Error analysis (Analysis of 50 samples).......... 14
9. Model 1 - Regression on top of BERT..........14
10. Model 2 - Regression on top of BERT with labels.......... 14
Conclusion..........15

1. Choosing the Dataset

The data should include Wikipedia text. The following options were examined:

- wikimedia (wikipedia dumps) - I considered extracting all the abstracts to use as data. It contains 27 files, ~6.5 GB each - so it is too big and long to process.
- Wikipedia (Huggingface) - consist of a preprocessed and cleaned subset of the wikimedia data. contains in total 6,458,670 entries of full texts.
- WikiText-103 - This dataset consists of 863,572 paragraphs extracted from Wikipedia. It has two versions: one is preprocessed and clean, but contains masked words. The other version is its raw version. It seems to be the right size.

Final decisions:

- Use the raw version of WikiTest-103 (Since we want to mask the words ourselves).
- More advantages of the dataset over other existing (and smaller) alternatives can be found in the "Reasons for a New Dataset" section of this paper. A major advantage is that the dataset was gathered specifically for research on named entities. This means that the data collection focused on issues that are important for named entities. For example, named entities are not common in randomly selected sentences from Wikipedia, so it is crucial to sample paragraphs instead.
- The WikiTest-103 dataset is separated into train, dev, and test sections. I decided to combine it and intend to split it again later because not all sentences are suitable for our needs. For example, some sentences may not contain two or more relevant named entities.
- If more data will be needed, it can be easily sampled from Huggingface’s Wikipedia.

2. Data Cleaning

link to code
link to drive
The dataset escapes dots and commas associated with numbers, and hyphens, with the "@" symbol. Additionally, the text was tokenized using the Moses tokenizer. When the dataset is loaded, I reversed these two processes to assist with Spacy's NER extraction process and also to provide suitable input for the BERT model, which does not expect a tokenized text. As this process is very time consuming, I performed it only once using the designated data_preprocessor.py script.

A sample of a raw entry from the dataset as taken from the WikiText-103 dataset's site [note: this is one entry in the dataset representing a paragraph]

```
= Super Mario Land =

 Super Mario Land is a 1989 side @-@ scrolling platform video game , the first in the Super
Mario Land series , developed and published by Nintendo as a launch title for their Game
Boy handheld game console . In gameplay similar to that of the 1985 Super Mario Bros. , but
resized for the smaller device 's screen , the player advances Mario to the end of 12
levels by moving to the right and jumping across platforms to avoid enemies and pitfalls .
Unlike other Mario games , Super Mario Land is set in Sarasaland , a new environment
depicted in line art , and Mario pursues Princess Daisy . The game introduces two Gradius
@-@ style shooter levels .
 At Nintendo CEO Hiroshi Yamauchi 's request , Game Boy creator Gunpei Yokoi 's Nintendo R
& D1 developed a Mario game to sell the new console . It was the first portable version of
Mario and the first to be made without Mario creator and Yokoi protégé Shigeru Miyamoto .
Accordingly , the development team shrunk Mario gameplay elements for the device and used
some elements inconsistently from the series . Super Mario Land was expected to showcase
the console until Nintendo of America bundled Tetris with new Game Boys . The game launched
alongside the Game Boy first in Japan ( April 1989 ) and later worldwide . Super Mario Land
was later rereleased for the Nintendo 3DS via Virtual Console in 2011 again as a launch
title , which featured some tweaks to the game 's presentation .
 Initial reviews were laudatory . Reviewers were satisfied with the smaller Super Mario
Bros. , but noted its short length . They considered it among the best of the Game Boy
launch titles . The handheld console became an immediate success and Super Mario Land
ultimately sold over 18 million copies , more than that of Super Mario Bros. 3 . Both
contemporaneous and retrospective reviewers praised the game 's soundtrack . Later reviews
were critical of the compromises made in development and noted Super Mario Land 's deviance
from series norms . The game begot a series of sequels , including the 1992 Super Mario
Land 2 : 6 Golden Coins , 1994 Wario Land : Super Mario Land 3 , and 2011 Super Mario 3D
Land , though many of the original 's mechanics were not revisited . The game was included
in several top Game Boy game lists and debuted Princess Daisy as a recurring Mario series
character .
```

More information about the dataset and its original usage can be also found in this youtube clip discussing the paper.

Reference: Merity, Stephen, et al. "Pointer sentinel mixture models." arXiv preprint arXiv:1609.07843 (2016).

3. Dataset Creation

3a. Data Processing and Filtering

link to code
link to drive
I used Spacy for extracting the named entities from the texts. I found this process to be very time-consuming. In order to speed up this process, I did the following:

- **Load all the sentences and process them as a group, rather than loading and processing each sentence separately** - when a spacy doc is created, it requires getting and releasing a number of resources that by default are not reused between calls. Therefore, calling doc once for all texts speeds up the processing (Reference).
- **Remove irrelevant components for the NER task from the Spacy pipeline** - such as tok2vec or lemmatizer.
- **Use parallelization** - use 4 processes for Spacy processing (applied according to the Spacy API).
- **Save Spacy's processing output locally for further computation** - to avoid the need to redo the processing each time.

Note: Spacy model used: en_core_web_lg

Another consideration that was taken into account is removing irrelevant named entities. Here is a table summarizing the Named entities labels extracted and used:

Entity label	**Used?**
PERSON: People, including fictional.	Yes
NORP: Nationalities or religious or political groups.	No
FAC: Buildings, airports, highways, bridges, etc.	Yes
ORG: Companies, agencies, institutions, etc.	Yes
GPE: Countries, cities, states.	Yes
LOC: Non-GPE locations, mountain ranges, bodies of water.	Yes
PRODUCT: Objects, vehicles, foods, etc. (Not services.)	Yes
EVENT: Named hurricanes, battles, wars, sports events, etc.	Yes
WORK_OF_ART: Titles of books, songs, etc.	Yes
LAW: Named documents made into laws.	Yes
LANGUAGE: Any named language.	Yes
DATE: Absolute or relative dates or periods.	No
TIME: Times smaller than a day.	No
PERCENT: Percentage, including "%".	No
MONEY: Monetary values, including unit.	No
QUANTITY: Measurements, as of weight or distance.	No

ORDINAL: "first", "second", etc.	No
CARDINAL: Numerals that do not fall under another type.	No

MI calculation

Mutual information (MI) measures the amount of information that is shared between two random variables. It is a non-negative quantity that is calculated based on the joint probability distribution of the variables.

To calculate the mutual information (MI) between two entities, e1 and e2, we need to define the relevant probabilities.
Let x be the probability of e1 appearing in a pair of named entities in a sentence.
We can represent x as a Bernoulli random variable, x ~ Bernoulli(p), and estimate p as follows: $p_{est}\,(x = 1) = \frac{\#\,pairs\ containing\ e1}{\#\,total\ pairs}$. $p_{est}\,(x = 0)$ is its complementary probability.
Similarly, we can apply the same estimation for e2 with y ~ Bernoulli(q).

To estimate the joint probability of x=1 and y=1, we use:
$P(x = 1,\ y = 1) = \frac{\#\,pairs\ containing\ both\ e1\ and\ e2}{\#\,total\ pairs}$. and so:

$$MI(e1, e2) = p(x = 0, y = 0) \cdot log \frac{P(x=0, y=0)}{p(x=0)p(y=p)} + p(x = 0, y = 1) \cdot log \frac{P(x=0, y=1)}{p(x=0)p(y=1)} + p(x = 1, y = 0) \cdot log \frac{P(x=1, y=0)}{p(x=1)p(y=0)} + p(x = 1, y = 1) \cdot log \frac{P(x=1, y=1)}{p(x=1)p(y=1)}$$

Reference: Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008. (link to relevant section in online book).

Addressing Single Entity Cases: the Null Entity Approach

To handle word entities appearing alone in a sentence, I introduced a dummy "null entity." For example, in the sentence "Rina likes Rex", the relationship between "Rina" and "Rex" is considered. However, in the sentence "Rina is nice", the relationship between "Rina" and the "null entity" is considered. This captures the fact that "Rina" can appear alone in a sentence, not always alongside "Rex".

PMI calculation

Pointwise mutual information (PMI) is a *variation* of mutual information that focuses on the relationship between individual occurrences or events rather than the overall distribution. It is commonly used in natural language processing and text mining to measure the association between two terms within a corpus of documents.

PMI is calculated as:

$$PMI(x, y) = log \frac{p(x,y)}{P(x) * P(y)}$$

where x and y are the specific occurrences, and p(x, y), p(x), and p(y) represent their respective probabilities.

Reference: Church, Kenneth, and Patrick Hanks. "Word association norms, mutual information, and lexicography." Computational linguistics 16.1 (1990): 22-29. (link)

Generated Dataset Structure

The generated dataset contains the following features:

- *sent_id* - unique generated id.
- *sent* - the original sentence (after preprocessing).
- *masked_sent* - *sent* with two sampled named entities masked.
- *ent1* - the text of the first masked named entity as appeared in the sentence.
- *label1* - the label of the first named entity (as identified by Spacy).
- *ent2* - the text of the second masked named entity as appeared in the sentence.
- *label2* - the label of the second named entity (as identified by Spacy).
- *mi_score* - the Mutual Information score of the two named entities.
- *pmi_score* - the Pointwise Mutual Information score of the two named entities.

Example for a single sample:

```
130,Du Fu's popularity grew to such an extent that it is as hard to measure his influence
as that of Shakespeare in England: it was hard for any Chinese poet not to be influenced by
him.,Du Fu's popularity grew to such an extent that it is as hard to measure his influence
as that of [MASK] in [MASK]: it was hard for any Chinese poet not to be influenced by
him.,Shakespeare,PERSON,England,GPE,9.34920129022248e-07,0.7647018015512669
```

Datasets and datasets' statistics

Throughout this project, I worked with two sets of data. One set, which I called "data500," was made using a rule that required entities to appear at least 500 times (K=500). The other set, named "data1000," followed a similar rule but with a higher count of at least 1000 occurrences (K=1000).

data500 statistics

Total lines extracted (containing one sentence or more): 863556
Entities >= 500 found: 1150
Number of sentences left after filtering entities < K: 321760
Dataset creation total time: ~1 hour.

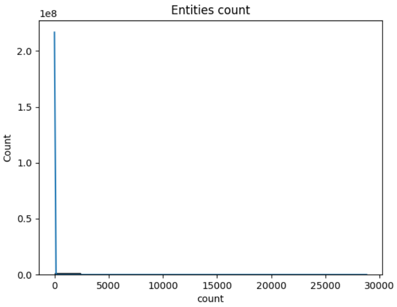
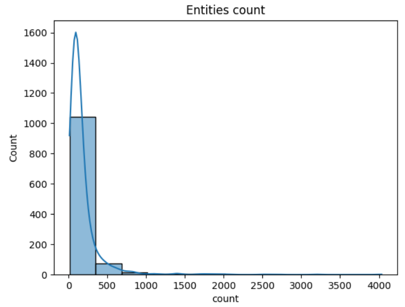
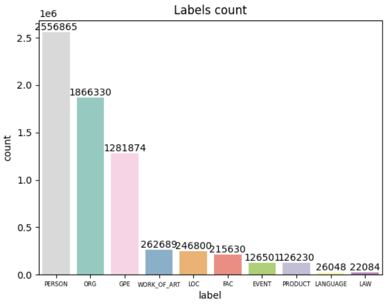
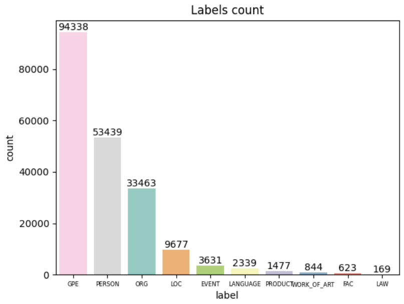
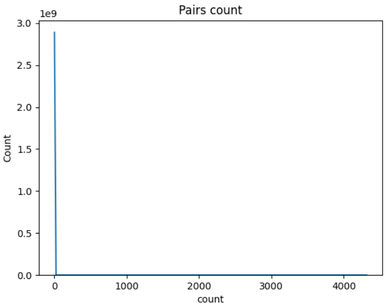
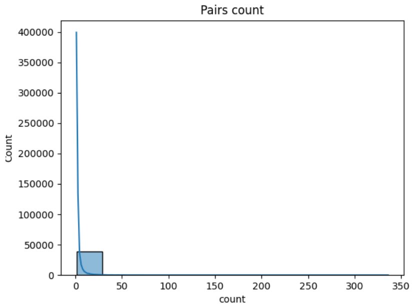
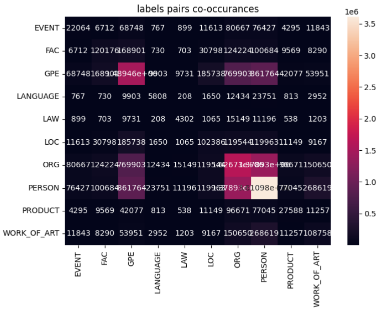
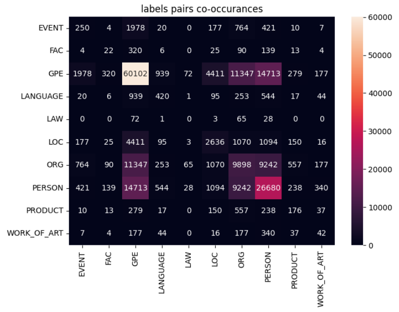
data1000 statistics

Total lines extracted (containing one sentence or more): 863556
Entities >= 1000 found: 484
Number of sentences left after filtering entities < K: 202756
Dataset creation total time: ~1 hour.

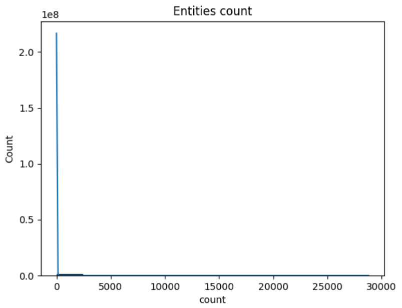
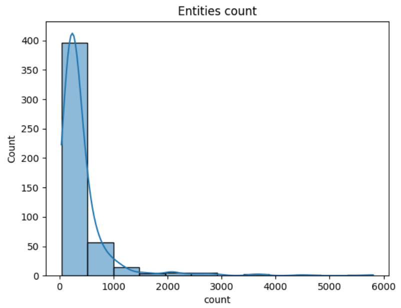
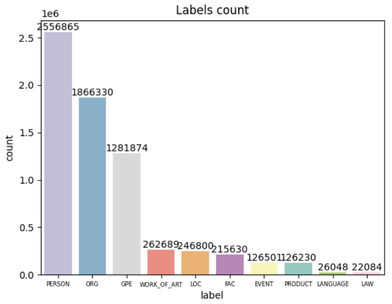
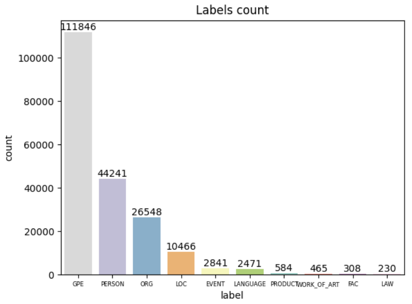
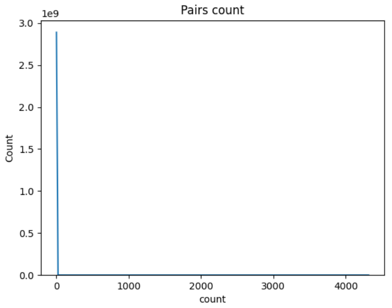
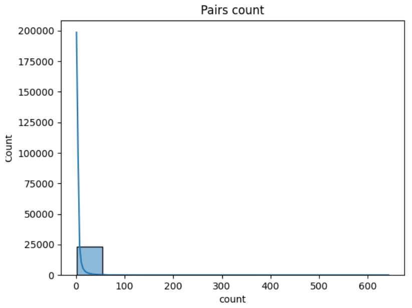
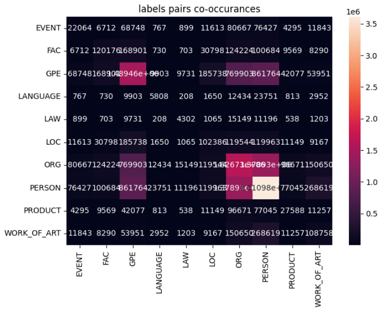
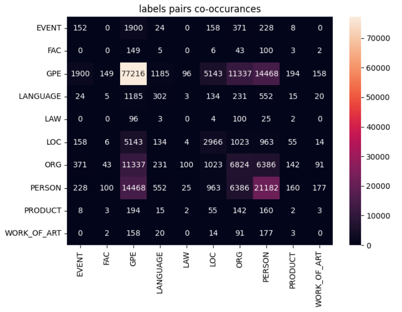
Comparing Data Distributions: Original vs. Sampled Dataset

Once sentences without at least 2 important names were removed, and a random selection of N=100,000 relevant sentences was made from the remaining ones, the dataset is ready. Here are the statistics of the sampled dataset, compared to those of the original dataset.

data500

	Original dataset - WikiText-103	Sampled dataset
Entities count		
Labels count		
Pairs count		
pairs-labels count		

data1000

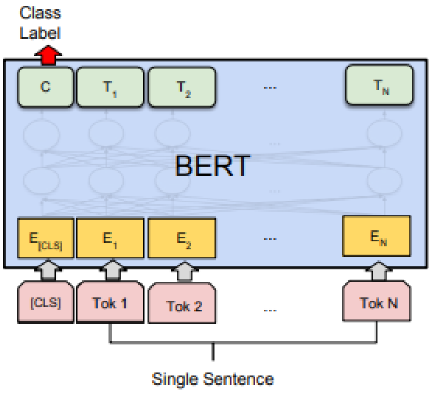
	Original dataset - WikiText-103	Sampled dataset
Entities count		
Labels count		
Pairs count		
pairs-labels count		

3b. Encoding Creation

link to code
link to drive

The dataset should also include the masked sentence encodings created by the BERT model. These encodings can be used later for regression and classification tasks. This will simulate the use of a "frozen" BERT model to encode sentences, followed by a custom deep network. It will significantly optimize the experiments, as the dataset encoding with BERT typically requires around 3 hours. Given its feasibility, it is preferable to undergo this process just once, rather than repeating it for each network training.

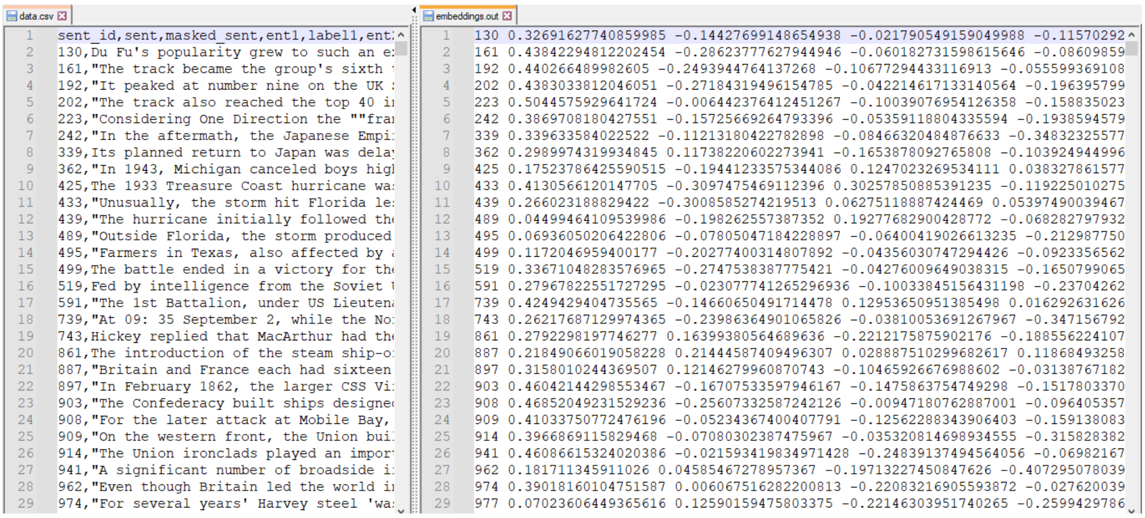
This approach also follows the recommended way of using the BERT model for classification.



A visual representation demonstrating the process of Single Sentence Classification Tasks using BERT, taken from the BERT research paper

I decided to separate the process of creating the encodings into its own section for two reasons: it is easier to use a separate Python script, and the output is quite large, so it will be saved in a separate file.

The structure of the encodings file is as follows: each line contains the ID of the sentence in the dataset file, followed by a space and a list of 768 numbers representing the BERT encoding of the sentence, with each number separated by spaces.



```
sent_id,sent,masked_sent,ent1,label1,ent:
130,Du Fu's popularity grew to such an e:
161,"The track became the group's sixth
192,"It peaked at number nine on the UK
202,"The track also reached the top 40 i
223,"Considering One Direction the ""fra
242,"In the aftermath, the Japanese Empi
339,Its planned return to Japan was dela
362,"In 1943, Michigan canceled boys hig
425,The 1933 Treasure Coast hurricane wa
433,"Unusually, the storm hit Florida le
439,"The hurricane initially followed th
489,"Outside Florida, the storm produced
495,"Farmers in Texas, also affected by
499,The battle ended in a victory for th
519,Fed by intelligence from the Soviet
591,"The 1st Battalion, under US Lieuten
739,"At 09: 35 September 2, while the No
743,Hickey replied that MacArthur had th
861,The introduction of the steam ship-o
887,"Britain and France each had sixteen
897,"In February 1862, the larger CSS Vi
903,"The Confederacy built ships designe
908,"For the later attack at Mobile Bay,
909,"On the western front, the Union bui
914,"The Union ironclads played an impor
941,"A significant number of broadside i
962,"Even though Britain led the world i
974,"For several years' Harvey steel 'wa
```

```
130 0.32691627740859985 -0.14427699148654938 -0.021790549159049988 -0.11570292
161 0.43842294812202454 -0.28623777627944946 -0.060182731598615646 -0.08609859
192 0.440266489982605 -0.2493944764137268 -0.10677294433116913 -0.055599369108
202 0.4383033812046051 -0.27184319496154785 -0.042214617133140564 -0.196395799
223 0.5044575929641724 -0.006442376412451267 -0.10039076954126358 -0.158835023
242 0.3869708180427551 -0.15725669264793396 -0.05359118804335594 -0.1938594579
339 0.339633584022522 -0.11213180422782898 -0.08466320484876633 -0.3483232557
362 0.2989974319934845 0.11738220602273941 -0.1653878092765808 -0.103924944996
425 0.17523786425590515 -0.19441233575344086 0.124702326953411 0.038327861577
433 0.4130566120147705 -0.3097475469112396 0.30257850885391235 -0.119225010275
439 0.266023188829422 -0.3008585274219513 0.06275118887424469 0.05397490039467
489 0.04499464109539986 -0.198262557387352 0.19277682900428772 -0.068282797932
495 0.06936050206422806 -0.07805047184228897 -0.06400419026613235 -0.212987750
499 0.1172046959400177 -0.20277400314807892 -0.04356030747294426 -0.0923356562
519 0.33671048283576965 -0.2747538387775421 -0.04276009649038315 -0.1650799065
591 0.27967822551727295 -0.023077741265296936 -0.10033845156431198 -0.23704262
739 0.4249429404735565 -0.14660650491714478 0.12953650951385498 0.016292631626
743 0.26217687129974365 -0.23986364901065826 -0.03810053691267967 -0.347156792
861 0.279229819774627 0.16399380564689636 -0.2212175875902176 -0.18855622410
887 0.21849066019058228 0.21444587409496307 0.028887510299682617 0.1186849325
897 0.315801024436950 7 0.12146279960870743 -0.10465926676988602 -0.0313876718
903 0.46042144298553467 -0.16707533597946167 -0.1475863754749298 -0.1517803370
908 0.46852049231529236 -0.25607332587242126 -0.00947180762887001 -0.096405357
909 0.41033750772476196 -0.05234367400407791 -0.12562288343906403 -0.159138083
914 0.396686911582946 8 -0.07080302387475967 -0.035320814698934555 -0.315828382
941 0.46086615324020386 -0.021593419834971428 -0.24839137494564056 -0.06982167
962 0.18171134591102 6 0.04585467278957367 -0.19713227450847626 -0.407295078039
974 0.39018160104751587 0.006067516282200813 -0.22083216905593872 -0.027620039
977 0.07023606449365616 0.12590159475803375 -0.2214630395174026 5 -0.2599429786
```

Reference: Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

4. Exploratory Data Analysis (EDA)

Link to Jupyter notebook
The EDA process is performed and documented directly in the Jupyter notebook itself.
The notebook in the link makes use of *data1000* dataset.
Please refer to it for more information.

5. Model 1 - Classification on top of BERT

link to code
link to drive - experiments results (experimenter.py output)

The purpose of this section is to develop a deep network model that can identify whether a sentence contains a relationship between two named entities. The model assigns a label of 1 when a connection exists and 0 when it does not. The labels are determined based on a threshold set using the MI or PMI median score.
The input for the network is the sentence encoded with BERT, where the two entities are masked. To make the training and experimentation process more efficient, we will use the BERT encodings created in part 3b. These encodings were generated once and took a lot of time and resources to create. Now, we can use them multiple times to train and evaluate different networks.
The main focus in this part is on refining the network by examining changes in the layer structure, learning rate and batch-sizes. Various thresholds and configurations were experimented with, as shown in the table below:

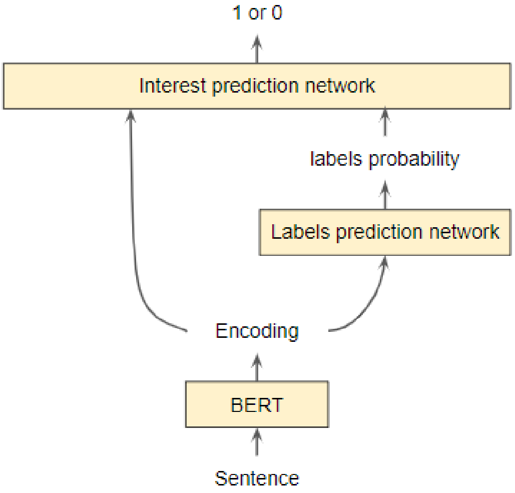
Calibration parameter	Tested values
Score	MI or PMI
Learning rate	0.01, 0.05, 0.001, 0.005
Batch size	256, 128, 64
Interest-prediction network architecture	2-4 layers uniformly sampled from the following dimensions: [64, 128, 256, 512]
Epochs	40

For each possible configuration, the loss, accuracy and F1 score for the train, validation, and test sets were collected. The runtime for each configuration was logged as well.

6. Model 2 - Classification on top of BERT with labels

link to code
link to drive - experiments results (experimenter.py output)

In this part, we will enhance the network developed in the previous part by incorporating additional information that has not been addressed: the labels of the masked Named Entities within the sentence. The network will first attempt to identify the relevant labels, and only then will the classification process be executed, considering both the predicted labels types and the sentence itself. Following is an illustration of the network:



For that end, we'll include an additional row in the original configurations table:

Calibration parameter	Tested values
label-prediction network architecture	2-4 layers uniformly sampled from the following dimensions: [64, 128, 256, 512]

7. Baseline models

link to code
link to drive - experiments results (experimenter.py output)

To ensure that the networks trained in sections 4 and 5 have truly learned valuable information rather than just basic rules, we will evaluate the performance of 3 models that classify based on simple heuristics.
* **Input for the models:** The masked sentence (this is what the BERT model received as input).
* **Models' output:** Classification (either 1 or 0).
* **The 3 heuristics tested**:

1. Entity Count: Count the number of entities in the sentence. If the count surpasses a certain threshold, the classification will be 1; otherwise, it will be 0. All possible thresholds will be tested.
2. Mask Distance: Measure the number of tokens between the two tokens containing the MASK placeholder in the sentence. If this distance exceeds a certain threshold, the classification will be 1; otherwise, it will be 0. All possible thresholds will be tested.
3. Random Score: Assign a random value of 0 or 1 to each sentence. This score will serve as the sentence's classification.

8. Classification Models Performances - Conclusions

8a - Models comparison

The results are displayed in the table below:

	Model 1		Model 2		Entity Count Heuristic		Mask Distance Heuristic		Random Score Heuristic	
dataset	max val accuracy	max test accuracy	max val accuracy	max test accuracy	max val accuracy	max test accuracy	max val accuracy	max test accuracy	max val accuracy	max test accuracy
data500	64%	63.6%	64.2%	64.1%	50%	50.3%	50.2%	50.5%	50.2%	50.3%
data1000	65.6%	65%	65.9%	64.9%	50.1%	49.9%	50.1%	50.8%	49.5%	50.5%

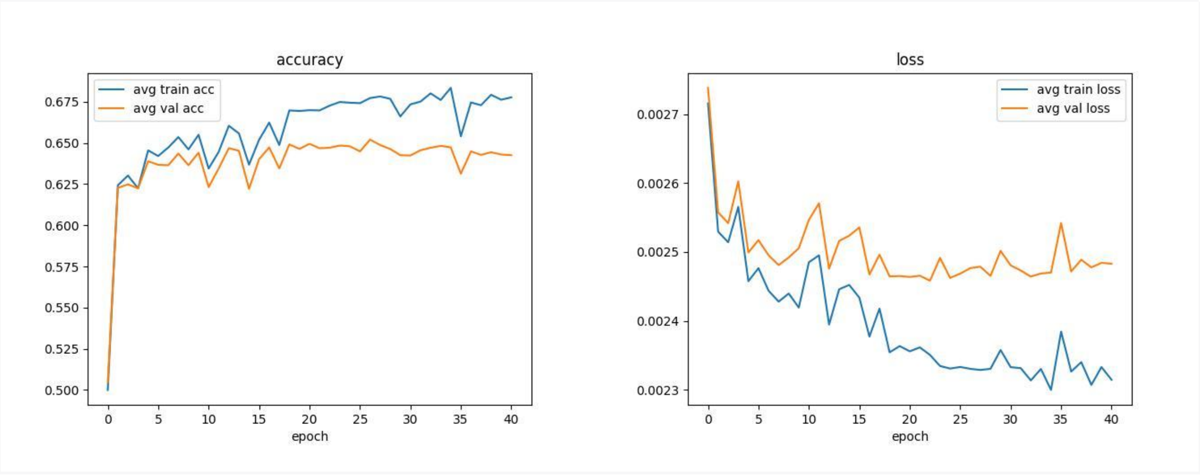
Baseline models' accuracy percentages varied from 40% to 50% (maximum achieved: 50.5%), demonstrating that the models 1 and 2 did, in fact, learn a complex function for prediction, rather than a basic, trivial one.

When we look at the dataset, the scores are better with data1000 than with data500. Also, there isn't a big difference between using PMI or MI because both give similar results.

The top accuracy that model 2 achieved for label prediction on the validation dataset (for both datasets) was around 53.6%. It's important to note that the entities labels are not binary.

As for the model, it's clear that model 2 doesn't make a big improvement compared to model 1. Overall, considering its simplicity and comparable runtime performance, it appears that model 1 is preferable over model 2.

Here is an example of a learning and loss graph from experiment 30 for model 1 and dataset data1000, which achieved the highest tested accuracy score:



Note that regularization techniques and early stopping need to be addressed - I chose to use a large number of epochs solely for the experiments.

8b Discovering the Optimal Configuration for Network Efficiency

Now, let's tackle the question of finding the most efficient way to set up the network. Our goal is to use as few layers and neurons as possible, while still getting good results.

When we look at the results privded in this link, which show 12 different setups using *data1000*, we see that just one layer is enough. Even if it only has 64 neurons, it still gives us pretty good accuracy rate – about 65.5% accuracy for the validation set and 64.8% accuracy for the test set. But, there's an important setting we need to consider - the *learning rate*: It's better to keep this rate low, to help the network learn slowly and steadily.

8c Error analysis (Analysis of 50 samples)

The detailed analysis can be found in this notebook.
Please refer to it for more information.

9. Model 1 - Regression on top of BERT

link to code
link to drive - experiments results (experimenter.py output)

This part employs the same Model 1 setup as in Part 5. However, this time, it will be used for a different task: instead of attempting to perform classification based on the MI/PMI score, it aims to predict the actual MI/PMI score of the masked entities pair itself.
Note: This model was only tested with the data500 dataset.

10. Model 2 - Regression on top of BERT with labels

link to code
link to drive - experiments results (experimenter.py output)

This part uses the same Model 2 setup as in Part 6 but for a regression task as well. This regression task will consider both the prediction for the labels of the masked entities and the encoded sentence to predict the MI/PMI score of the entities pair.
As with the previous classification task, according to the results, this model variation doesn't improve the performance of model 1.
Note: This model was only tested with the data500 dataset.

Conclusion

This work demonstrates the ability to predict significance of the relationships between entities in sentences. Its conclusions and programming framework serve as a foundation for future research that should delve deeper into examining the functions the model has learned, fully analyzing and mapping the model's errors, and experimenting with further modifications to the input and structure of the networks to improve the learning results.