Natural Language Processing for Fact-Checking and Claim Assessment

Intermediate report of project advancement

Othman EL HOUFI
Dimitris KOTZINOS

MSc Research in Data Science & Machine Learning

December 17, 2021

Abstract

As false information and fake news are propagating though out the internet and social networks, the need of fact-checking operations becomes necessary in order to maintain a truthful digital environment where general information can be reliably exploited whether in politics, finance and other domains. The need of this online claim assessment comes from the fact that fake news and false information can have a big negative impact on politics, economy (2016 USA Elections) and public health (COVID-19).
A number of solutions have been proposed to deal with this problem and limit the spread of false information, both manual and automatic. Of course the manual approaches done on websites such as *PolitiFact.com*, *FactCheck.org* and *Snopes.com* don't construct a viable solution for the long term as the speed and scale of information propagation increase exponentially rendering this manual fact-checking operation where human fact-checkers can't scale up at the same rate limited and incapable of solving the problem.
Here, we present our contribution in this regard: an automated solution for fact-checking using Wikipedia's articles for claim verification. The algorithm uses NLP techniques in order to extract the so-called claim from the user input, then, using Wikipedia's API, it retrieves all the relevant articles and assesses with a degree of confidence if the claim is true, false or unable to decide due to lack of information showing evidence (sentences in articles) and probabilities for each resulted case.

Keywords: Natural Language Processing, Wikipedia, Information retrieval, Text processing, Natural Language Inferencing, Fact-Checking, Document retrieval, Sentence retrieval, Fake-news.

Contents

Chapter 1

Introduction

1.1 Project Context

From a social and psychological perspective, humans have been proven irrational and vulnerable when differentiating between truth and false news (typical accuracy ranges between 55% and 58%), thus fake news obtain public trust relatively easier than truthful news because individuals tend to trust fake news after repeated exposure (*Validity effect*), or if it confirms their pre-existing beliefs (*Confirmation bias*), or simply due to the obligation of participating socially and proving a social identity (*Peer pressure*). The social sciences are still trying to comprehend the biological motivations that makes fake news more appealing to humans.

On the other hand, the growth of social media platforms resulted in a huge acceleration of news spreading whether true or false. As of Aug. 2017, 67% of Americans get their news from social media. These platforms even give the user the right to share, forward, vote and participate to online discussions. All of this made the problem of fake news spreading more and more dangerous, our economies for example, are not robust to the spread of falsity, false rumors have affected stock prices and the motivations for large-scale investments, as we witnessed after a false tweet claimed that Barack Obama was injured in an explosion which caused $130 billion drop in stock value. Another recent example is related to public health where rumors about COVID-19 vaccines and drug companies influenced people in their decision on getting vaccinated.

That being said, is there a way to monitor the spread of fake news through social media? Or more specifically, how can we differentiate between fake news and truthful news, and at what level of confidence can we do that?

From a computer engineering perspective, different approaches were studied:

- **Knowledge-based Fake News Detection:** a method aims to assess news authenticity by comparing the knowledge extracted from to-be verified news content with known facts, also called fact-checking.
- **Style-based Fake News Detection:** focuses on the style of writing, i.e. the form of text rather than its meaning.
- **Propagation-based Fake News Detection:** a principled way to characterize and understand hierarchical propagation network features. We perform a statistical comparative analysis over these features, including micro-level and macro-level, of fake news and true news.
- **Credibility-based Fake News Detection:** the information about authors of news articles can indicate news credibility and help detect fake news.

In this project we will focus on the method of **Knowledge-based Fake News Detection** also called **Fact-Checking**. The goal is not to implement an algorithm that scans social networks for real time fake news detection, but rather we will create a model that can assess with a degree of confidence the truthfulness or falseness of a claim given by a user as an input by exploiting Wikipedia's articles as a source of true knowledge and export evidence that validates or refutes the subjected claim.

1.2 Use case scenario

Suppose that while browsing the internet or talking to people you come across a claim that says *"The former U.S president John F. Kennedy died in September 22, 1963"*, as it is a general truth and not a relative truth it should be easier to verify the validity of this claim as well as find evidence that proves it.
Using the platform we will create, you can simply write the claim you like to verify with no regards to a specific linguistic rule, the model will extract relevant articles from Wikipedia using an API, then it retrieves sentences relative to your claim and apply a comparison in order to assess if the claim is True, False, or Not Enough Information as well as giving a percentage of confidence and evidence of the results that were processed straight from Wikipedia's database.
Combing back to our example, the model should return that *"John F. Kennedy died in November 22, 1963"* so the input claim is false.

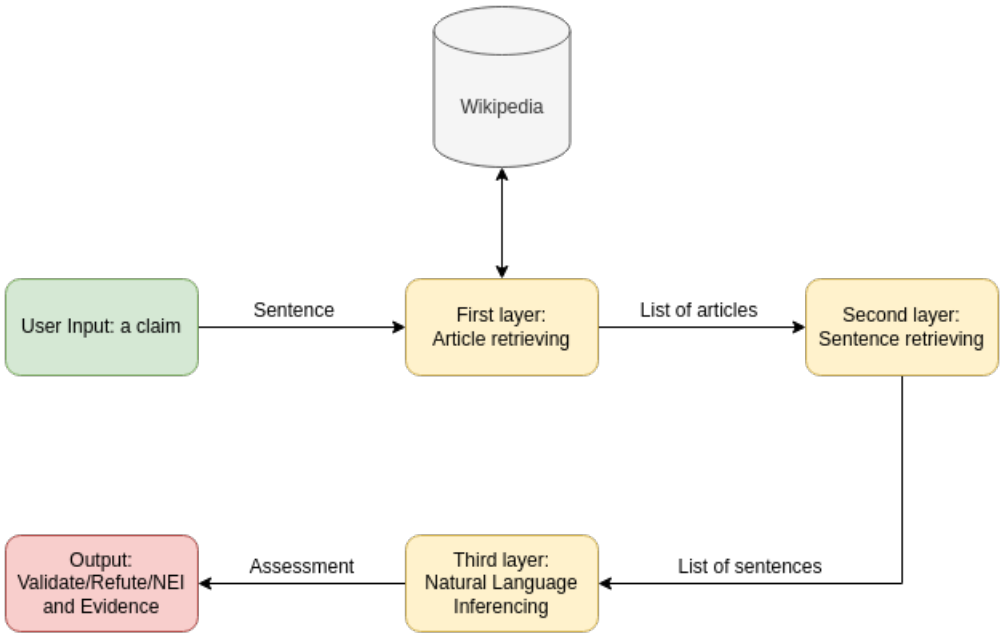


Figure 1.1: General Model Processing Pipeline

Chapter 2

Identified challenges and solutions

2.1 Fact-Checking challenges

In order for the model to work each layer/part of the system must answer to a specific task, the combination of results made by each layer constructs a robust model that is able with a degree of confidence to fact-check a claim as well as present evidence of the assessment. Although for each layer to work as intended we must find solutions to these challenges:

- **Claims Spotting:** the model must be robust to linguistic changes, we must deal with different phrasing for the same or similar claims. For every "reasonable" input we must extract the target claim.
- **Articles Retrieving:** as Wikipedia holds millions of articles, the model must look for a limited number of relative articles to the input claim with an order of degree of correlation.
- **Time Recording:** relative articles in Wikipedia can be outdated, for example Britain belongs to the European Union is an outdated knowledge. The model must be sensitive to the timestamps of articles.
- **Sentences Retrieving:** in each article retrieved from Wikipedia's database we must extract sentences relative to our input claim in order to apply a "kind of" comparison as well as present the winning sentences as evidence to the user.
- **Sentence Comparison:** here we must create a model or use a pre-existing natural language inferencing model in order to compare retrieved sentences from Wikipedia's articles and the input claim.
- **Credibility Evaluation:** not all informations in Wikipedia are true.

These can be regarded as main challenges of our Fact-Checking project, but, evidently, other problems can be presented for example the verifiability of claims, not all claims can be verifiable, especially if it is an personal opinion or a personal belief, in this case the model must not give a True or False assessment but it should tell the user that there is not enough information (NEI) to make such an assessment.

2.2 Related work and solutions

2.2.1 Styled based fake news detection

A study done by *Piotr Przybyła* named *Capturing the Style of Fake News* in 2020 from Institute of Computer Science, Polish Academy of Sciences, in order to detect fake news or in other words assess the credibility of an article by looking at the style of writing rather than the meaning of the words and sentences.
The purpose of this study was to prove that general-purpose text classifiers, despite their good performance when evaluating simplistically, they overfit to sources of documents in training data. In contrast to this method, a truly style-based prediction that uses an analysis of the stylometric model shows that it focuses on sensational and affective vocabulary, known to be typical for fake news.

Fake news sources usually attempt to attract attention for short-time financial or political goal (Allcott and Gentzkow 2017) rather than to build a long-term relationship with the reader, in this perspective, the language used by these sources tend to be informal, sensational and affective (Bakir and McStay 2017). This can be used to build a classifier for indicating low credibility.

First of all they started by gathering a corpus of 103,219 documents from 223 online sources labeled by media experts like *PolitiFact* and *Pew Research Center*. Then they designed two models: a neural network and a model based on features used in stylometric analysis. This has a purpose to demonstrate that the stylometric features based model captures the affective language elements.

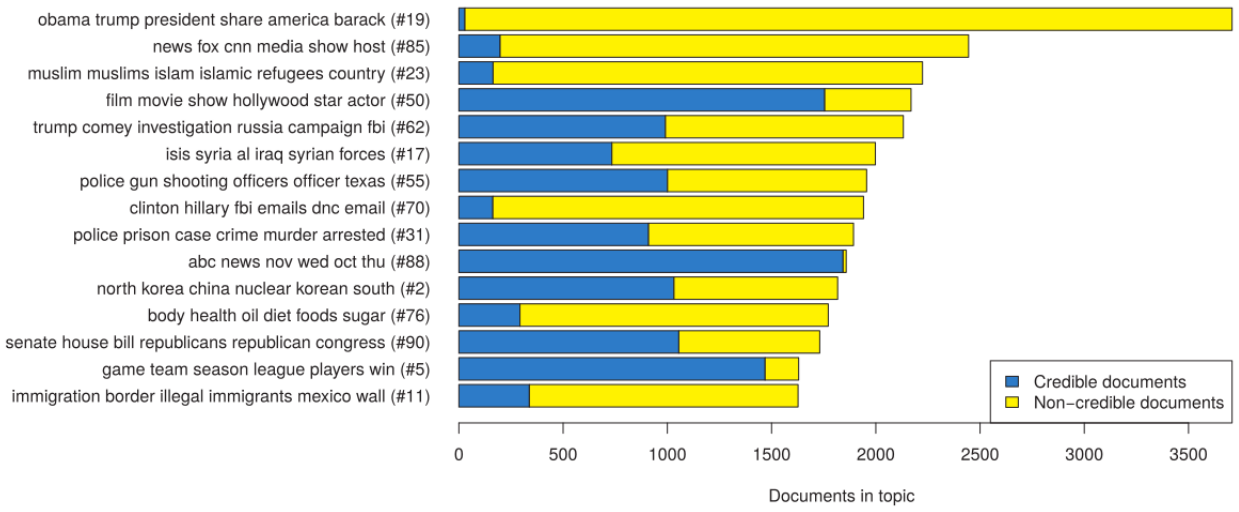


Figure 2.1: From Piotr Przybyła paper - The largest 15 LDA (Latent Dirichlet Allocation) topics in the corpus, each shown with the six most significant keywords, an identifier and bars illustrating number of credible and non-credible documents associated with it

Stylometric classifier: the architecture of this classifier is generally based on a collection of stylystic features followed by a linear modeling. The features are:

- number of sentences, average sentence length (in words) and average word length (in characters),
- number of words matching different letter case schemes (all lower case, all upper case, just first letter upper case, other), represented as counts normalized by the document length,
- frequencies of POS uni-grams, bi-grams and trigrams, represented as counts normalized by the document length (if present in at least 5 documents),
- frequencies of words belonging to the 182 word categories in the expanded GI dictionary, represented as counts normalized by the document length.

Neural network classifier: the applied solution was BiLSTMAvg a neural network with architecture based on elements used in natural language processing, i.e. word embeddings (Mikolov et al. 2013) and bidirectional LSTM (Hochreiter and Schmidhuber 1997). The following layers are included:

- An embedding layer, representing each token using a 300-dimensional word2vec vector trained on Google News,
- Two LSTM layers, forward and backward, representing each sentence by two 100-dimensional vectors (output of the last cell in a sequence),
- A densely-connected layer, reducing the dimensionality to 2 and applying softmax to compute class probability,
- An averaging layer, representing each document's class probability scores by averaging the scores for all its sentences.

The neural network is implemented and trained in TensorFlow for 10 epochs with sentence length limited to 120 tokens and document length limited to 50 sentences.

In order to understand if general-purpose text classifiers can capture document style without overfitting to features like the source or topic of document and to compare with the two classifiers they created, they evaluated two baseline models: bag of words and BERT.

The evaluation protocol consisted on running the model learning and prediction in a 5-fold cross validation (CV) scenario and comparing it's output to target labels. They used accuracy as a metric of evaluation rather than precision or recall.

Method	doc. CV	topic CV	source CV
Stylometric	0.9274	0.9173	0.8097
BiLSTMAvg	0.8994	0.8921	0.8250
Bag of words	0.9913	0.9886	0.7078
BERT	0.9976	0.9965	0.7960

Figure 2.2: From Piotr Przybyła paper - Classification accuracy of our stylometric and neural classifiers compared to baselines in three evaluation scenarios, simulating, respectively, a new document from known sources and topics, a document from unknown topic and a document from unseen source

The obtained results shows that stylometric based classifier loses 10% of the accuracy on unseen sources even if it has more consistent performance over evaluation scenarios. Piotr Przybyła explains this drop in accuracy by assuming that the model specialize in the style of individual sources rather than the general style of fake news. Nevertheless, he was able to prove that his model takes into account the affective words in order to classify fake news.

2.2.2 Knowledge graph based fake news detection

Studying the *"DEAP-FAKED"* paper published in 2021 by *Mohit Mayank*, *Shakshi Sharma* and *Rajesh Sharma*, it turned out that by exploiting techniques related to network analysis, Natural Language Processing (NLP) and the implementation of Graph Neural Networks (GNNs) we can detect fake news with an F1 score of 88%.

The *"DEAP-FAKED"* consists of three individual components:

- **News encoder:** this component performs the contextual encoding of news title. trying both unidirectional and bidirectional sequence encoders, they implemented a 2-layer stacked biLSTM as the main subcomponent the news encoder.
- **Entity encoder:** this component identifies the named entities present in the news title and encodes the individual entities using knowledge graph (KG). For example, a news title with text *"US Officials See No Link Between Trump and Russia"*, contains two entities *"Trump"* of person type and *"Russia"* of geolocation type. The framework includes relevant entities in the news then encodes them. They used Wikidata, an open-source KG, as the source to match the entities and ComplEx KG embedding technique to embed entities.
- **Classification Layer:** this component consolidates the news encoder's and entity encoder's representations to perform the final downstream Fake News classification learning. In this framework, the two representations are concatenated to create a super representation of the news content and entities. This representation is then passed to further non-linear activated layers with decreasing layer dimensionality. The final layer consolidated the information into a single dimension which is activated by a sigmoid layer, where the final output represents the probability of the news as either true or fake.

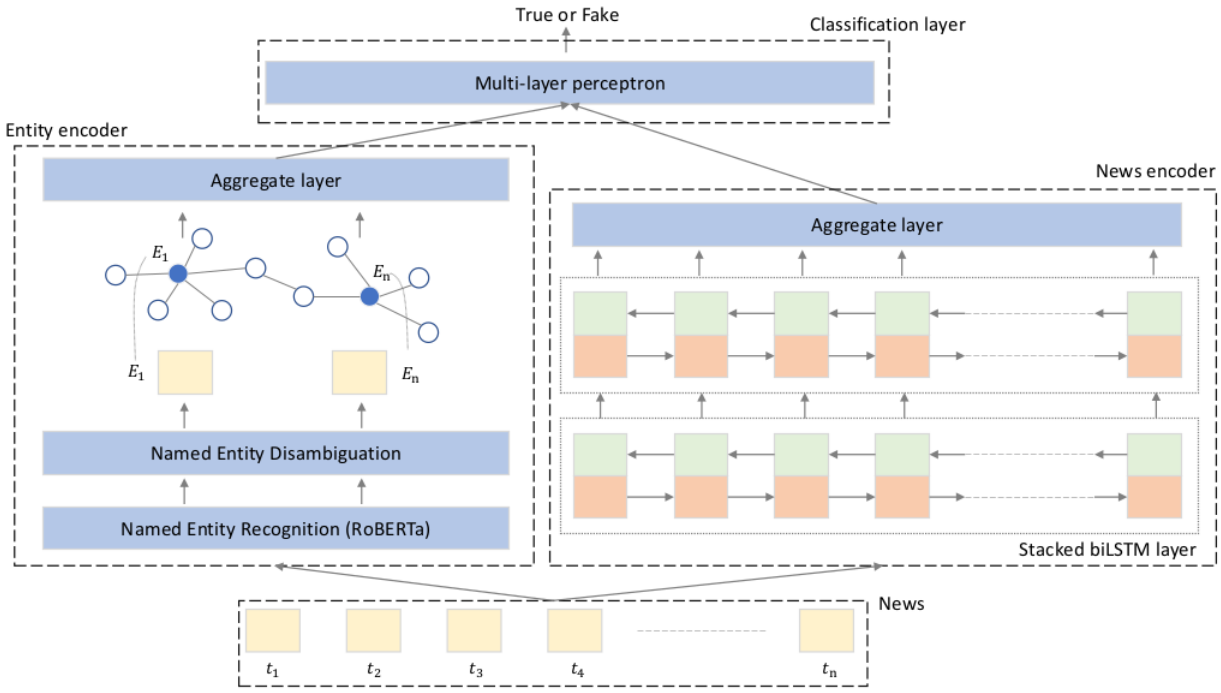


Figure 2.3: From DEAP-FAKED paper - Illustration of the proposed DEAP-FAKED framework. The left sub-figure shows the entity encoder module. The right sub-figure shows the news encoder module. The top sub-figure is the Fake News classifier module. The bottom shows the tokenized news as the input to the entity encoder and news encoder.

The dataset used for *DEAP-FAKED* framework can be described as following:

- **Fake News dataset:** The first dataset is the Kaggle Fake News dataset, which consists of 20,387 news items, having a near equal combination of true and Fake News. The news covers several domains such as Politics, Business, and Technology. The complete pre-processing step brought down the news item count to ∼14k with a distribution of 60% - 40% of true and Fake News classes, respectively.They called this dataset KFN-UB. The second dataset is CoAID, which contains diverse COVID-19 healthcare misinformation, including Fake News from websites and social platforms. CoAID includes 4,251 news items. After pre-processing it only 632 news item and called CoAID-UB.
- **Knowledge Graph:** the framework uses Wikidata5M as a knowledge graph, it is created by only considering the "valid" facts, where the validity is confirmed if all entities and relations in the fact have a Wikipedia article and long description.

As for the evaluation procedure, they chose five baseline models for comparison with *DEAP-FAKED* framework:

1. ExtraTreeClassifier: a decision tree-based classification algorithm that fits randomized decision trees on various sub samples of the dataset. In this case they first extract the count vectorization-based feature matrix from the tokenized news items and then pass it to ExtraTreeClassifier.
2. LSTM: Long-Short Term Memory is a gated variant of Recurrent Neural Networks. It mitigates the exploding gradient problem of the classical sequence-based neural networks. In this case they pass the title of the tokenized news items to the LSTM layer and connect the last hidden state of the LSTM with a sigmoid activated MLP layer to perform the Fake News classification.
3. SentRoBERTa: a modification of the pre-trained RoBERTa network that uses siamese and triplet network structures to derive semantically meaningful sentence embeddings. In this case they use SentRoBERTa to generate sentence level embedding for news titles that are connected with the sigmoid activated MLP layer to perform the Fake News classification.

4. StackedBiLSTM: a two layer stacking of the conventional bidirectional LSTM layer. In this case they pass the tokenized news title to the StackedBiLSTM layer and connect the last hidden state with sigmoid activated MLP layers to perform the Fake News classification.

5. EntWiki-StackedBiLSTM: This model incorporates the entities along with the news title. For each news item, the news title is encoded using the StackedBiLSTM, as discussed before. They extract the Wikipedia description of the entity identified in the news item and encode that description using SentRoBERTa. The news title and entity encoding are then concatenated and passed to the sigmoid activated MLP layer to perform the Fake News classification.

	KFN-UB				CoAID-UB			
Model vs Dataset	**F1 avg.**	**F1 std.**	**Acc avg.**	**Acc std.**	**F1 avg.**	**F1 std.**	**Acc avg.**	**Acc std.**
ExtraTreeClassifier	0.7663	0.002	0.7831	0.002	0.7526	0.009	0.7638	0.008
LSTM	0.7810	0.009	0.8109	0.008	0.7255	0.021	0.7402	0.008
SentRoBERTa	0.6476	0.054	0.6879	0.040	0.7292	0.130	0.7375	0.116
StackedBiLSTM	0.7878	0.005	0.8137	0.002	0.7476	0.046	0.7585	0.030
EntWiki-StackedBiLSTM	0.8809	0.006	0.8898	0.008	0.7436	0.031	0.7454	0.032
DEAP-FAKED	**0.8866**	0.007	**0.8955**	0.007	**0.7813**	0.032	**0.7822**	0.033

Figure 2.4: From DEAP-FAKED paper - Performance score of the models detailed in the paper is presented here. For each of the dataset, we report F1 macro and Accuracy metric values. We present average and standard deviation of the performance observed after performing 3 trials with different starting seed. For both the datasets, DEAP-FAKED reports the best performance value.

As evident by the results, *DEAP-FAKED* reports the highest score on both of the datasets.

2.2.3 Hierarchical Propagation based fake news detection

In a paper wrote by *Kai Shu*, *Deepak Mahudeswaran*, *Suhang Wang*, and *Huan Liu* with the name of *Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation*, the detection of fake news can be done by looking at the correlation between news and the hierarchical characteristics of a social network.

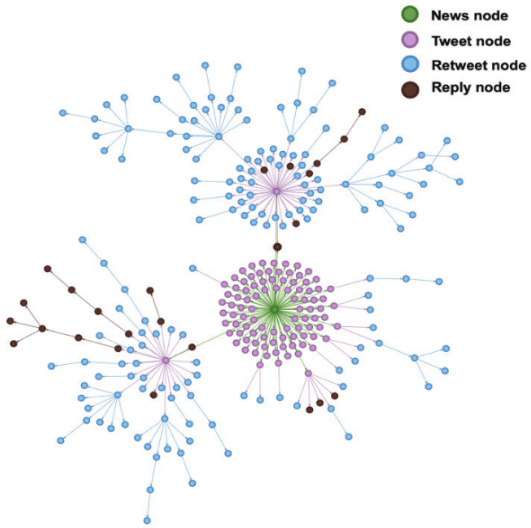
The datasets used for this study were the public fake news repository *FakeNewsNet* (Shu et al. 2017) that contains data related to different fact-checking websites that offers news content, social context and dynamic information. In addition, they used the data from fact-checking websites *GossipCop* and *PolitiFact* that contains news articles with labels annotated by professional journalists. News content includes meta attributes of the news like body text, he social context includes the related user interactions with the news like posting, sharing, commenting on Twitter, and dynamic information contains the timestamps of user's interactions.

After collecting all of this data, building a hierarchical propagation network was the next step. First of all they defined two major levels of the hierarchy:

- **Micro-level:** involves users conversations towards news pieces on social media over time. It contains rich information of user opinions towards news pieces.
- **Macro-level:** encompasses information on tweets posting pattern and information sharing pattern. It represents the global propagation of information through a cascade of re-tweets.

Platform	PolitiFact	GossipCop
# True news	624	16,817
# Fake news	432	5,323
# True news with propagation network	277	6,945
# Fake news with propagation network	351	3,684
# Users	384,813	739,166
# Tweets	275,058	1,058,330
# Retweets	293,438	530,833
# Replies	125,654	232,923

(a) The statistics of FakeNewsNet



(b) An example of the hierarchical propagation network of a fake news piece fact-checked by Politifact. It consists of two types: macro-level and micro-level. The macro-level propagation network includes the news nodes, tweet nodes, and retweet nodes. The micro-level propagation network indicates the conversation tree represented by cascade of reply nodes.

Figure 2.5: From the *Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation* paper

In both these levels they used *Structural Analysis* to identify patterns in conversations, *Temporal Analysis* to understand the exchange of opinions in terms of time. For the Micro-level they added a *Linguistic Analysis* to inference the sentiment of people around news pieces, thus obtaining linguistic features for statistical comparison.

To evaluate the performance of fake news detection of the model they choose randomly 80% of news articles for training and the remaining 20% for testing. The results shows high scores for both datasets:

Datasets	Level	Acc	Prec	Rec	F1
PolitiFact	Micro	0.834	0.823	0.847	0.835
	Macro	0.807	0.816	0.789	0.802
	All	**0.843**	**0.835**	**0.851**	**0.843**
GossipCop	Micro	0.843	0.841	0.845	0.843
	Macro	0.852	0.841	0.868	0.854
	All	**0.861**	**0.854**	**0.869**	**0.862**

Figure 2.6: From the *Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation* paper - Evaluation of the model

2.2.4 Perplexity based fake news detection

In March 2021, *Nayeon Lee*, *Yejin Bang*, *Andrea Madotto*, *Madian Khabsa*, and *Pascale Fung* published a paper called *Towards Few-Shot Fact-Checking via Perplexity* where they propose a new approach of the powerful transfer learning ability of a language model via a perplexity score. Using a method called *few-shot learning*, they built a model that outperforms major class baseline models by more than 10% on the F1-Macro metric score.

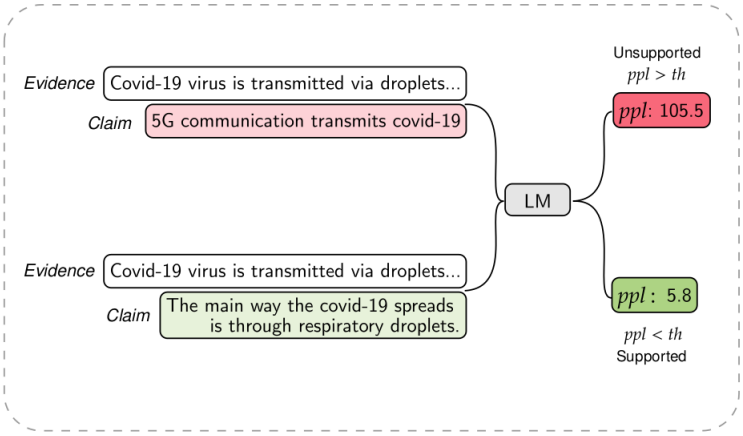


Figure 2.7: From the *Towards Few-Shot Fact-Checking via Perplexity* paper - Illustration of our simple yet effective perplexity-based approach. Few-shot data samples are used to find the optimal perplexity threshold th that separates Unsupported claims from Supported claims.

Few-Shot Learning refers to the technique of feeding a machine learning model with a very small amount of training data to guide its predictions, like a few examples at inference time, as opposed to standard fine-tuning techniques which require a relatively large amount of training data for the pre-trained model to adapt to the desired task with accuracy.
In NLP, *Few-Shot Learning* can be used with Large Language Models (LMs), which have learned to perform a wide number of tasks implicitly during their pre-training on large text datasets. This enables the model to generalize, that is to understand related but previously unseen tasks, with just a few examples.

Perplexity is a commonly used metric for measuring the performance of LMs. It is defined as the inverse of the probability of the test set normalized by the number of words:

$$PPL(X) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{p(x_i|x_0, \ldots, x_{i-1})}}.$$

Figure 2.8: Perplexity score

Another way of interpreting perplexity is as a measure of the likelihood of a given test sentence with reference to the training corpus.

In this paper the goal is to determine the veracity of a claim given some evidence, for this they define a claim, evidence pair. The label *Supported* is assigned when relevant evidence exists that supports the claim, and *Unsupported* label for the opposite case.
Unsupported claims on average have higher perplexity than *Supported* claims. For example, *Supported* claim "Washing hands prevents the spread of diseases" has a perplexity value of 96.74, whereas the *Unsupported* claim "All dogs speak English fluently" has a much higher perplexity value of 328.23.

Unsupported Claims	Perplexity	**Supported** Claims	Perplexity
5G network can spread diseases.	826.70	Beyonce is one of the most famous singers in the world.	23.03
All dogs speak English fluently.	328.23	Chicago is one of the cities in the United States.	43.92
Washing hands **helps** the spread of diseases.	201.10	Washing hands **prevents** the spread of diseases.	96.74

Figure 2.9: From the *Towards Few-Shot Fact-Checking via Perplexity* paper - Relations between veracity of claim and perplexity. Unsupported claims have higher perplexity compared to Supported claims. Note that the perplexity score listed here is using GPT2-base on each of the claims.

The datasets used in this experiment are:

- **Covid19-Scientific:** a new test set constructed by collecting COVID-19-related myths and scientific truths labeled by reliable sources like MedicalNewsToday, the Centers for Disease Control and Prevention (CDC), and the World Health Organization (WHO). The set contains 172 claims with labels *(Supported, Unsupported)* obtained from the aforementioned reliable sources.
- **Covid19-Social:** a collection of 340 COVID-19-related claims fact-checked by journalists from Politifact.com. Unlike the Covid19-Scientific dataset, it contains non-scientific and socially-related claims, such as "For the coronavirus, the death rate in Texas, per capita of 29 million people, we're one of the lowest in the country".
- **FEVER:** Fact Extraction and Verification (FEVER) is a publicly released large-scale dataset generated by altering sentences extracted from Wikipedia to promote research on fact-checking systems.

For the models created for this perplexity based experiment they used one unidirectional LM and one masked LM:

- PPL_{GPT2-B} : a single-parameter classifier based on perplexity from GPT2-base (Radford et al., 2019) (unidirectional LM)
- PPL_{BERT-B} : a single-parameter classifier based on perplexity from BERT-base (Devlin et al., 2019) (Masked LM)

As for baseline models they finetuned pre-trained Transformer-based (Vaswani et al., 2017) in order to build a classifier, which is a common approach used to achieve many state-of-the-art results in the literature.

Moving to the experimental setup, they used the *Few-Shot* technique in order to validate or train the models. Given N_D as the size of the dataset D, we do an n-shot experiment with n samples from D as a "validation set" for our perplexity-based approach or as a "training set" for the fine-tuning approach, and the remainder $(N_D - n)$ as a test set. For example in the 2-shot experiment using the Covid19-Social dataset (340 samples), we have two training samples and 338 test samples.
They took into consideration the accuracy and the F1-Macro metrics for the evaluation. Because the datasets are unbalanced, they mainly consider the F1-Macro score over accuracy as an overall evaluation.

Shot #	Models	Fine-tuning?	Size	Covid-Scientific		Covid-Social		FEVER	
				Acc	F1-Macro	Acc	F1-Macro	Acc	F1-Macro
	Major Class	N/A	N/A	58.72%	37.00%	77.35%	43.62%	50.00%	33.33%
2	BERT-B_{ft}	yes	110M	47.34%	32.21%	26.11%	23.33%	51.56%	37.34%
	BERT-L_{ft}	yes	336M	49.39%	34.80%	37.78%	27.81%	50.80%	36.49%
	RoBERTa$_{ft}$	yes	125M	52.66%	34.29%	40.75%	26.78%	50.00%	33.33%
	XLNet$_{ft}$	yes	110M	51.48%	48.49%	57.67%	44.35%	49.41%	44.65%
	PPL_{GPT2-B}	no	117M	**66.75%**	**64.39%**	62.61%	**53.61%**	**61.92%**	**57.50%**
	PPL_{BERT-B}	no	110M	47.93%	38.54%	**77.74%**	49.15%	52.54%	41.33%
10	BERT-B_{ft}	yes	110M	46.27%	31.70%	43.26%	30.70%	51.56%	37.34%
	BERT-L_{ft}	yes	336M	50.00%	36.74%	60.49%	42.18%	50.80%	36.49%
	RoBERTa$_{ft}$	yes	125M	52.64%	40.28%	40.73%	26.73%	50.00%	33.33%
	XLNet$_{ft}$	yes	110M	49.69%	42.44%	59.68%	39.45%	49.41%	44.65%
	PPL_{GPT2-B}	no	117M	**72.98%**	**68.57%**	**71.23%**	**55.11%**	**62.82%**	57.04%
	PPL_{BERT-B}	no	110M	63.15%	60.77%	61.90%	46.35%	57.59%	**57.11%**
50	BERT-B_{ft}	yes	110M	56.75%	53.61%	60.21%	36.91%	52.18%	38.82%
	BERT-L_{ft}	yes	336M	56.75%	39.15%	64.94%	44.07%	51.14%	39.99%
	RoBERTa$_{ft}$	yes	125M	56.40%	38.97%	73.13%	45.30%	50.44%	38.15%
	XLNet$_{ft}$	yes	110M	63.22%	51.98%	**77.62%**	43.70%	49.18%	48.42%
	PPL_{GPT2-B}	no	117M	**74.73%**	**73.83%**	73.63%	**59.91%**	**67.48%**	**64.70%**
	PPL_{BERT-B}	no	110M	62.53%	61.11%	71.11%	54.72%	57.44%	56.94%

Figure 2.10: From the *Towards Few-Shot Fact-Checking via Perplexity* paper - Results comparison among perplexity-based classifiers and fine-tuned classifiers in 2-shot, 5-shot and 10-shot settings across three different tasks. Models whose names start with PPL are our proposed perplexity-based classifiers. Major Class is a reference to evaluate classifier performance. All test results reported are mean values of three trials with randomly selected n-shot training samples from the dataset, where $n = 2, 10, 50$.

We can observe that the perplexity-based classifiers, especially PPL_{GPT2-B}, outperform all Major Class baselines across all tasks in all settings. For instance, PPL_{GPT2-B} outperforms the Major Class by a great margin of 16% and 36.8% on accuracy and F1-Macro scores. Evidence conditioned perplexity scores are capable of providing signals regarding the veracity of the the given claim.

2.2.5 Credibility based fake news detection

Chapter 3

Conclusion

Ces dernières années, de nombreuses techniques différentes de création d'empreintes digitales et d'indexation ont été proposées et sont maintenant utilisées dans des produits commerciaux. Dans ce projet, nous avons examiné de plus près l'une de ces techniques, qui a été développée à l'origine pour le système d'identification audio *Shazam*. Nous avons discuté des idées principales qui sous-tendent ce système, mais il y a de nombreux paramètres qui doivent être ajustés afin de trouver un bon compromis entre les différentes exigences, notamment la robustesse, la spécificité, l'évolutivité et la compacité. Les aspects importants sont les suivants :

- les paramètres de la STFT (longueur de la fenêtre, taille du saut) qui déterminent les résolutions temporelle et spectrale,
- la stratégie de sélection et d'extraction des pics spectraux (avec ses paramètres de voisinage),
- la taille des zones cibles (utilisées pour définir les triplets), et
- des structures de données appropriées pour le hachage.

Bien que ce système est robuste à de nombreux types de distorsions du signal, l'approche de création d'empreinte discutée n'est pas conçue pour gérer les déformations temporelles. La correspondance des cartes de constellation ainsi que les différences d'horodatage (*timestamp*) dans les paires de pics sont toutes deux sensibles aux différences de tempo relatif entre la requête et le document de base de données. Par conséquent, il est nécessaire d'utiliser d'autres techniques pour être invariant aux modifications de l'échelle de temps.

Les empreintes digitales utilisant les pics spectraux sont conçues pour être très sensibles à une version particulière d'un morceau de musique. Par exemple, face à une multitude d'interprétations différentes d'une chanson par le même artiste, le système d'empreintes digitales est susceptible de choisir la bonne, même si elles sont pratiquement indiscernables par l'oreille humaine. En général, les systèmes d'identification audio sont conçus pour cibler l'identification d'enregistrements qui sont déjà présents dans la base de données. Par conséquent, ces techniques ne sont généralement pas généralisables aux enregistrements en direct ou aux performances qui ne font pas partie de la base de données.

Chapter 4

Perspectives

Comme décrit avant, il y a de nombreux paramètres qui doivent être ajustés afin de trouver un bon compromis entre les différentes exigences, notamment la robustesse, la spécificité, l'évolutivité et la compacité. Trouver des valeurs optimales à ces paramètres pourra augmenter largement les performances de notre application du point de vue de la robustesse aux distorsions, voire aussi du point de vue la mémoire utilisée et la vitesse de recherche d'une correspondance. Or, ce n'est pas une simple tâche, de plus les paramètres de notre application augmente, le processus de trouver des valeurs optimales à ces paramètres devient très compliqué.

Parmi les solutions que nous envisageons comme extension à notre application est l'utilisation d'un modèle de réseau neurones artificielles qui prendra en entrée les paramètres de notre application, et la sortie sera divisée sur les différentes exigences voulues tel que la robustesse, la mémoire, et le temps de recherche.

Ce réseau sera entraîné sur une large base d'apprentissage qui provienne de plusieurs tests déjà effectués d'une manière dynamique, par exemple nous allons exécuter la reconnaissance des morceaux sur une large collections de musiques tout en ajoutant du bruit et d'autres distorsions et aussi en variant le temps d'enregistrement du microphone, les résultats obtenus feront une très bonne base d'apprentissage pour notre réseau de neurones artificielles. Peut-être même on pourra ajuster les paramètres de notre application dynamiquement par rapport à chaque situation.

Bibliography