"Authorship Verification" A Natural Language Processing case study

Gabriele Pisciotta
g.pisciotta1@studenti.unipi.it
Student ID: 596217

Fabio Murgese
f.murgese@studenti.unipi.it
Student ID: 583714

Miriana Somenzi
m.somenzi@studenti.unipi.it
Student ID: 604009

ABSTRACT

In this report we investigate different approaches for the task of "Authorship Verification", highlighting differences in terms of performances. This task, originally presented at PAN 2020 conference, consists in the evaluation of different fictional texts in order to see if they have been written by the same author using Natural Language Processing techniques, which involve also neural approaches.

KEYWORDS

Natural Language Processing, Authorship Verification

1 INTRODUCTION

Authentication is a major concern in today's global information society; the task of Authorship Verification in particular is something on which the Natural Language Processing community has been working since the early days.
With the continuous growth of available data, especially thanks to Online Social Network (OSN) such as *Facebook* and *Twitter*, the need for automatic processes that help you certify the authorship of a text has gained a renewed interest. Recognizing the importance of developing methods that can help the *Open Source Intelligence* (OSINT) community to cope with the growing amount of textual material without a claimed and verified authorship coming from OSN like *Telegram* and *Reddit* is crucial, especially if you also take into account all the problems coming from *Identity Theft* and the cyber-security risks involved. In these circumstances understanding the identity of the person behind the text can be decisive, thinking about cases of online propaganda related to jihad and global terrorism [21].
To tackle this issue, we worked on a dataset consisting of a list of known authors who have written about a given set of topics and their texts; the test dataset contains verification cases from a subset of the authors and topics found in the training data[1].
The data we used comes from the PAN 2020 competition, and, as stated in their documentation, the train and test datasets consist of pairs of (snippets from) two different fanfictions, that were obtained from the website fanfiction.net. Each pair was assigned a unique identifier and distinguished between same-author pairs and different-authors pairs. They also offered metadata on the fandom (i.e. thematic category) for each text in the pairs (note that fanfic "crossovers" were not included and only single-fandom texts were considered).
The fandom distribution in the training dataset maximally approximates the (long-tail) distribution of the fandoms in the original dataset. The test dataset is structured in the exact same way, however a significant shift in the relation between authors and fandoms should be expected.

One of the main problems that these kind of systems have to overcome, as evidenced in literature, is the problem of overfitting: usually these models are tailored on very specific datasets and it's difficult to develop ones that do not overfit on specific content [15]. In fact, lot of work has been done within a *single topic condition*: in this scenario all the text involved is related to a single topic and the stylistic content (and the lexicon) is handled according to this. Also, most works consider the scenario in which a text belongs to only one topic. Our case, instead, presents a cross-topic scenario (also defined as *cross-domain*), with all texts being of the type *fanfiction* but belonging to different domains.

For this reason we'll formally define the problem from a proper *Machine Learning* (ML) point of view, as a binary classification problem:

Definition 1 (Authorship Verification as Binary Classification [5]). *Authorship verification can be formalized as the task of approximating the target function $\phi : (D_k, d_u) \rightarrow \{T, F\}$, where D_k is a set of documents of known authorship by the same author and d_u is a document of questioned authorship.*

In other terms, the task will be to learn a classifier such that it maps to T (being the boolean value 1 the truth value) if the author of D_k is also the author of d_u, or it will map to F (being the boolean 0 the falsity value) otherwise. Because we're dealing with a cross-domain (or *cross-topic*) context, it's not suitable to consider D_k a *set* of documents, because there's no guarantee that the documents in the set are related to the same topic and with the same language.

[1]https://pan.webis.de/clef20/pan20-web/author-identification.html

To simplify, we make the assumption of $k = 1$, making D_k a *singleton*, so a set with pairs of documents, each one having only one author: basically, our aim is to check pairs of documents, making the task more suitable for modeling from a ML point of view.

Furthermore, we consider useful to contextualize the task of Authorship Verification in the set of similar tasks that deal with text's authorship. Here, we summarize the main ones [14]:

- Authorship Attribution: given a document and a set of candidate authors, determine which of them wrote the document .
- Authorship Identification: automated identification of the individual(s) who authored an anonymous document on the basis of text-internal properties related to language and writing style .
- Authorship Profiling: trying to distinguish between classes of authors by studying how language is shared by people. This helps in identifying profiling aspects such as age, gender,and language variety, among others.
- Authorship Verification: given a pair of documents, determine whether they are written by the same author.
- Authorship Obfuscation: given a document and a set of documents from the same author, paraphrase the former so that its author cannot be identified anymore.
- Obfuscation Evaluation: devise and implement performance measures that quantify safeness, soundness, and/or sensibleness of an obfuscation software.

Having contextualized the problem, we propose the following experiments:

(1) A transformer based similarity computation
(2) A linguistic driven approach

All the experiments will be evaluated according to the following scores:

(1) AUC: the conventional area-under-the-curve score, as implemented in scikit-learn.
(2) F_1: the well-known performance measure (not taking into account non-answers), as implemented in scikit-learn.
(3) c@1 [18]: a variant of the conventional F1-score, which rewards systems that leave difficult problems unanswered (i.e. scores of exactly 0.5).
(4) $F_{0.5u}$ [6]: a newly proposed measure that puts more emphasis on deciding same-author cases correctly.

Further details will be described in the Section 4.

2 DATASET COMPOSITION

The dataset[7] is composed of pairs of snippets of fanfictions that were obtained from http://fanfiction.net, and made publicly available from PAN 2020 / PAN 2021 competition. It comes in two forms, the small and the large, with varying size of the number of examples.
As stated in the conference's guidelines[2], the first is most suitable for symbolic Machine Learning methods, while the second is tailored to be fed into Deep Learning models that need large amounts of data. It could also be interesting and challenging to cope with the smaller dataset and see the differences in terms of performance between the two.
Before going into details of the datasets, we recall the definition of fanfiction:

Definition 2 (Fanfiction [14]). *Fictional texts produced by non-professional authors in the tradition of a specific cultural domain (or 'fandom'), such as a famous author or a specific influential work*

The abundance of data is one of its major advantages, as fanfiction is nowadays estimated to be the fastest growing form of online writing; this particular style is also characterized by a relative wealth of author-provided metadata, relating to the textual domain (the fandom), period of production, and intended audience.

To what it concerns the texts, they've been normalized in order to avoid textual artifacts: this has been done by handling the punctuation and the white spaces.
For the dataset generation, the texts have been taken by author and fandom (that here acts as *topic*) in order to have a good mix of the two, despite the very uneven popularity of fandoms and activity of authors, to prevent gross overrepresentation of individual fandoms and authors, that could lead to dynamics of overfitting / underfitting.
For the large dataset, 148.000 same-author (SA) and 128.000 different-authors (DA) pairs were drawn from the fanfiction.net crawl. The SA pairs encompass 41.000 authors of which at least four and not more than 400 have written in the same fandom (median: 29). In total, 1.600 fandoms were selected and each single author has written in at least two, but not more than six fandoms (median: 2)[14].

As it has been said before, we're dealing with couples of documents. Each pair has been built according to all the possible $\binom{n}{2}$, with n being the number of texts from this author, without allowing two pairs with the same author and fandom.
The DA pairs were built from texts of 250.000 authors of

[2]https://pan.webis.de/clef21/pan21-web/author-identification.htmlrelated-work

doc 1	doc 2	same author	fandom
I shift a bit, warily letting my eyes dart fro...	"All will become one with Russia," he said, al...	True	Guardians of Ga'Hoole, Hetalia
"Told you." Jay walked over to the group of si...	smiled then whispered, "Can never say no to Ma...	False	Hocus Pocus, Boondock Saints

Table 1: Example took from dataset

which at least two and not more than 800 (twice the number, since each pair consists of two authors now) have written in the same fandom (median: 51). The number of fandoms is the same and largely overlaps with the SA pairs. Each author has written texts in at least one and not more than three fandoms (median: 1).
The small training set is a subset of the large training set with 28.000 SA and 25.000 DA pairs from the same 1,600 fandoms, but with a reduced author number of 6,400 (4–68 per fandom, median: 7) and 48.500 (2–63 per fandom, median: 38), respectively.

3 STATE OF THE ART

In this section we are going to highlight the State Of The Art (SOTA) of this task, according to the participants to the PAN 2020 conference.
It's possible to see a brief summary [14] in Tab. 2. The models involved spans from classical textual similarity to the newest neural approaches such as Attention mechanism and Convolutional Neural Networks (CNN), with also attempts to exploit data compression techniques to solve the task.
Different kind of features have been involved, highlighting the fact that feature engineering plays a role in these competition even with the newest neural models. Many attempted to involve fandom-related features, while others tried a more agnostic approach using stylometric features (e.g.: word frequencies, POS tagging, N-grams, ...), and also ensembles of classic features such as token or character n-grams representation. Also an approach of neural feature extraction has been tried.

As for the resulting performance for all the listed works, it's possible to see Tab. 3. As noticeable, the Tab. 3 has much more submissions than the works described in Tab 2. This is because Niven and Kao (niven20) and Faber and van der Ree (faber20) are not described in the conference's paper [14], so they are not included in Tab. 2, and in the meanwhile three of the teams (boenninghoff20, araujo20and weerasinghe20) submitted two systems, calibrated on the small and the large training datasets, respectively.

4 EXPERIMENTAL SETUP

Our experiments have been carried out on a server shared with our colleagues, having a GPU "NVIDIA Tesla P100 PCIe" with 16GB of memory and CPU "Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz" with 80 cores. We developed our interactive scripts for this task in Python, specifically using Jupyter Notebook to easily interact with each component of what developed.

The datasets of the Autorship Verification competition come in two different versions: we decided to consider only the "small" version, being it of 2 GB of texts, for time and resources constraints. Without considering the time spent for the training and testing of the models, even the data manipulation has been time and memory consuming: we first built a Pandas dataframe from the single JSON dump of the dataset, we splitted the full dataset in design set, and test set (75%, 25%), with the design set splitted again in training set and validation set (75%, 25%), exploiting the scikit-learn utilities to ensure reproducibility, by setting proper "random states" (seeds).

Models

We decided to compare different models:

(1) a Cross-Encoder model
(2) a set of linguistic features based models (reproduction and extension of the `weerasinghe20-small` work)

Definition 3 (Cross-Encoder [19]). *It's a transformer based model architecture composed of a BERT layer capable of embedding two sentences together and a classifier layer that finally gives an output* $\in [0, 1]$*, as described in Fig. 1.*

A Cross-Encoder is a particular type of Neural Network that doesn't produce an embedding for each sentence, that are instead passed together to the BERT transformer to get a single context vector in which they are combined. At the final stage this context is fed into a classifier layer to get a score representing the similarity between the two sentences. For our particular case study, in fact, we're not interested in extracting an embedding for each sentence, but we're rather interested in considering a representation space in which we're able to easyl discriminate between text written by the same author or not. We tried to answer to the following question: *is it possible to train a Cross-Encoder able to discriminate same author's texts without specific linguistic features to consider?*

Paper	Features	Model
Boenninghoff et al. [8]	Neural Features Extraction through Metric Learning	Siamese Networks and Probabilistic Linear Discriminant Analysis
Weerasinghe et al. [22]	Stylometric Features (function word frequency, POS tag N-grams, noun/verb phrases)	Logistic Regression Model and Neural Network
Halvani et al.[11]	Ensembles of punctuation and token n-grams, masked token n-grams, sentence starters and endings	Classification using the Manhattan metric
Kipnis, A. [16]	Higher Criticism Statistic and Similarity Score	Unsupervised approach
Gagala, L. [10]	Context-free Grammar preprocessing for PPM	Data Compression based on Prediction by Partial Matching (PPM)
Ikae, C. [12]	Relative frequency of most frequent words and punctuation marks for document representation	Text Similarity
Ordonez et al.	Text and Fandom-Specific Features	Long-sequence Transformer and Character based Convolutional Neural Network with self-attention layers
Araujo-Pino et al. [3]	Character N-gram representation	Siamese Neural Network

Table 2: State Of The Art [14]

Submission	AUC	C@1	$F_{0.5u}$	F_1	Overall
boenninghoff20-large	0.969	0.928	0.907	0.936	0.935
weerasinghe20-large	0.953	0.880	0.882	0.891	0.902
boenninghoff20-small	0.940	0.889	0.853	0.906	0.897
weerasinghe20-small	0.939	0.833	0.817	0.860	0.862
halvani20-small	0.878	0.796	0.819	0.807	0.825
kipnis20-small	0.866	0.801	0.815	0.809	0.823
araujo20-small	0.874	0.770	0.762	0.811	0.804
niven20-small	0.795	0.786	0.842	0.778	0.800
gagala20-small	0.786	0.786	0.809	0.800	0.796
araujo20-large	0.859	0.751	0.745	0.800	0.789
baseline (naive)	0.780	0.723	0.716	0.767	0.747
baseline (compression)	0.778	0.719	0.703	0.770	0.742
ordonez20-large	0.696	0.640	0.655	0.748	0.685
ikae20-small	0.840	0.544	0.704	0.598	0.672
faber20-small	0.293	0.331	0.314	0.262	0.300

Table 3: SOTA performances [14]

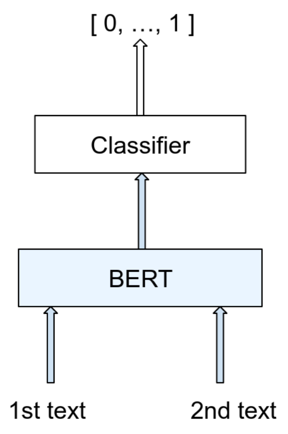

Figure 1: Cross-Encoder architecture

Cross-Encoder Model

We used a pretrained model then fine-tuned to our specific task. The model is taken from HuggingFace library: stsb-TinyBERT-L-4[13]. It's a distilled BERT transformer which involves Multi-Head-Attention mechanisms, with 4 layers, GeLU activation functions, dropout set at 0.1, a hidden size $d^{'}$ of 312, an intermediate size $d_i^{'}$ of 1200, 12 attention heads, and with a total of 14.5M parameters. It's been pretrained on the `STSbenchmark` dataset, coming from the tasks organized in the context of SemEval between 2012 and 2017, being it standard English based benchmark to compare among meaning representation systems in future years. We used for training purposes the utilities of the SBERT library, that make it faster to prototype and fine-tune pretrained models like ours. We used a batch size of 12 and trained it for 50 epochs, keeping the model that scored best on the validation set, according to the F_1 score. As for the tokenization, we used the Tokenizer pretrained with the stsb-TinyBERT-L-4.

`weerasinghe20-small`'s Model and its extensions

For what concerns the replication of the `weerasinghe20-small` model, we used the same kind of features used by the original work and we also extended them with other two features (Gunning Fox Readability index and the number of pronouns in the text). For time constraints, we used, in both tests, only 20k samples (half of the dataset) from the small dataset. We also extended the model including a Multilayer Perceptron classifier; for the same constraints described above we avoided trying a complete grid search and we decided to manually explore the different values for the hyperparameters' choice.

This works exploit the *Writeprints* set of features as follows (including our addition):

- Characters TF-IDF distribution: the distribution of different single characters in texts
- Word TF-IDF distribution: the distribution of different words, considering words as tokens divided from others by a space, in the different texts. In Figure 2 we printed the 30 most frequent tokens, excluding punctuation and function words, to have a rapid visualization of the subjects and writing style of the different authors

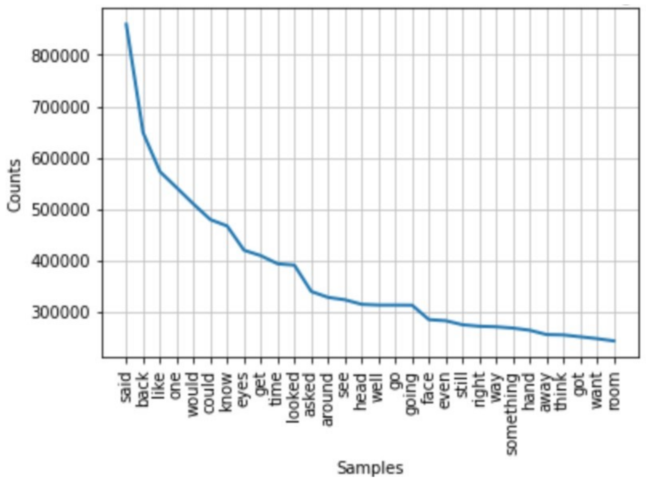

Figure 2: 30 Most common words

- Part Of Speech (POS) tag TF-IDF distribution: the distribution of the different parts of speech, with part of speech being the category assigned to the word in accordance with its synctactic function
- POS tag chunks TF-IDF distribution: distribution of pos tag chunks, with chunks being bigger units grouping various part of speech like noun phrases, verb phrases etc.
- POS tag chunks subtrees TF-IDF distribution
- Special chars distribution: distribution of special characters, with them being e.g.: @,*,£...
- Frequency of function words: frequence of appearence of function words, so words with little or no lexical meaning on their own that express grammatical relationships among content words e.g articles, quantifiers, prepositions
- Hapax Legomena: measure of vocabulary richness; in corpus linguistics an Hapax Legomenon is a word or expression which occurs only once in the entire context we are taking into consideration, so, in our case, is a word or expression that appears only once in the entire text
- Character count: number of single characters
- Character distribution per word: the distribution of single characters in words
- Average of character per word: average number of characters present in a word
- Word count: number of words
- Gunning Fox readability index: a readability index, where readability is intended as a measure of the understandability of a text, so an estimation of how difficult it is to read and comprehend. Readability is also strongly associated to lexical density, being it one of the factors that can determine the linguistic complexity of a written document. The Gunning Fox index is calculated on the average sentence length and the number of complex words present in the text, as we can see in the formula:
  $0.4 * [\frac{\text{total words}}{\text{total sentences}} + 100 * \frac{\text{complex words}}{\text{total words}}]$
- Number of pronouns: we decided to also consider just the pronouns based on the information found in [4], to try and have a glance of the authors' biological gender based on their choice of pronouns. We firstly tried an overall visualization, then we decide to discard the pronoun "you" because of its low informativeness, being both a singular and a plural pronoun. In Fig.3 and Fig.4 we can see a visualization of both the results

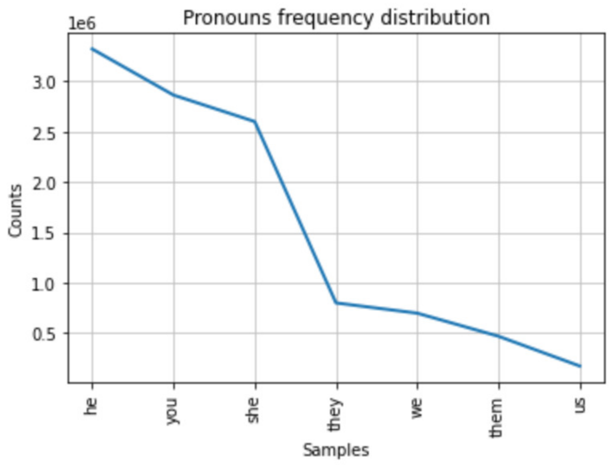

Figure 3: Pronouns distribution

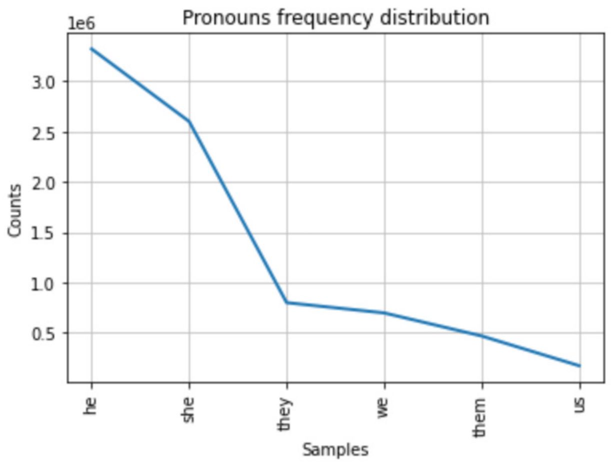

Figure 4: Pronouns distribution

Definition 4 (Writeprints). *The Writeprints method [1] is a method used in forensic linguistics to assess the authorship of text. Identity is established through the analysis of different features: lexical, syntactic, structural, idiosyncratic and content-specific ones.*

To classify each pair, we compute the features independently on the first and second text, and then we consider the absolute value of their difference ($|t_1 - t_2|$). In this way, we build a dataset in which each pair is represented by the differences of their features, and we then train a Logistic Regression classifier and different versions of the Multilayer Perceptron classifier.

Definition 5 (Logistic Regression [20]). *The logistic regression is a simple model described by the equation* $\frac{1}{1+e^{wx}}$, *in which we basically learn to binary separate the samples by softening the Perceptron's step-function with the continuous differentiable logistic equation previously described.*

Definition 6 (Multilayer Perceptron (MLP) [20]). *The Multilayer Perceptron, or feed-forward dense Neural Network, is a layered model made of different layers of artificial neurons, such that each unit receives input only from units in the immediately preceding layer. Each layer has a set of parameters* $\boldsymbol{W}$ *to be learned through a proper learning algorithm. Each unit computes a weighted sum of its input* $in_j = \sum_{i=0}^{n} w_{i,j} a_i$, *where a is the output of the previous layer, and then it applies an activation function g (e.g.: sigmoid) such that the output* $a_j = g(in_j) = g(\sum_{i=0}^{n} w_{i,j} a_i)$

As it's known, the Logistic Regression it's a simple model that perfectly fits for the task of binary classification, while the MLP can be a more expressive model depending on its architecture.

We therefore tried different MLP hyperparameters validating them and then showing the results on the test set for the model that better scored on the validation set. The hyperparameters explored are described in Tab. 4 For the

	hidden size	activation	solver	max iterations
Model 1	100	ReLu	Adam	20
Model 2	100	Logistic	Lbfgs	20
Model 3	200	Logistic	Lbfgs	50

Table 4: Model hyper-parameters explored

POS tagging, we used the Brill's treebank [9], for replication purposes. For the tokenization here we used the `NLTK`'s TreebankWordTokenizer, that uses regular expressions to tokenize text as in Penn Treebank.

5 RESULTS AND DISCUSSION

Cross-Encoder model

The Cross-Encoder scored the following, positioning ourselves just slightly below the PAN2020 baseline threshold:

	AUC	C@1	F_{05u}	F_1	overall
Cross-Encoder	0.74	0.696	0.706	0.727	0.719

Table 5: Cross-Encoder Test results

We see that the results found are not so impressive: we think that the chosen architecture doesn't properly fit the task, being it modelled for sentences similarity purposes and not entire texts' similarities. Also, TinyBERT has a known limitation of 512 token as the maximum sequence length, while our texts are obviously made of many more. A more specific "long-text" transformer would have probably been better (e.g.: *LongFormer*), but for resources limits we weren't able to train it. Even considering different pre-trained models (roberta-base and distilroberta-base), we weren't able to do better. A proper workaround, that could also be investigated as future work, would have been the splitting of each pair of text in many pairs of sentences, and the score related to each pair of text would have been the mean of the single score for each pair of sentences.

Weerasinghe20's model and its extensions

Concerning `weerasinghe20` based and extended models, we can see that, as in the original work of the author, the Linear Regression model performs slightly better than the MLP. Also, we noticed that using our extended set of features we were able to have a better performance, suggesting that expanding this works from a linguistic point of view can enhance the results in general, and this can be an interesting research direction.

	AUC	C@1	F_{05u}	F_1	overall
Log. reg. weerasinghe	0.914	0.914	0.912	0.914	0.913
Log. reg. extended	0.919	0.919	0.917	0.919	0.918

Table 6: Test Scores of the Logistic Regression models

	AUC	C@1	F_{05u}	F_1	overall
MLP 1 weerasinghe (val)	0.5	0.5	0.556	0.667	0.556
MLP 2 weerasinghe (val)	0.887	0.887	0.895	0.884	0.888
MLP 3 weerasinghe (val)	0.902	0.902	0.904	0.901	0.902
MLP 1 extended (val)	0.5	0.5	0.003	0.001	0.251
MLP 2 extended (val)	0.886	0.886	0.89	0.884	0.886
MLP 3 extended (val)	0.907	0.907	0.905	0.908	0.907
Best weerasinghe (test)	0.914	0.914	0.916	0.913	0.914
Best extended (test)	0.916	0.916	0.916	0.917	0.916

Table 7: Validation/Test Scores of the MLP models

For what it concerns the models trained using MLP, we can say that the general rule of applying the *Adam* solver for large datasets doesn't pay well in this case, having a better overall score for the model trained with the *Lbfgs* solver instead, that usually converges faster and performs better with smaller datasets. All scores can be seen in Tab. 7 and Tab. 6.

6 CONCLUSION

This task has been very challenging, from both a theoretical point of view and an implementative point of view. It has been interesting to test the two different approaches, a Transformer model based on the Cross-Encoder architecture and one that consider hand-engineered features related to the Writeprints method, which has been studied across these years for authorship profiling in forensics scenarios. The best model we discovered to be a Linear Regression classifier trained on engineered linguistic features.
According to the SOTA, most of the models presented an exploitation of specific features and only one model has been successful in exploiting neural features extraction.
We think that computational linguistics, and linguistics in general, plays a crucial role in the authorship verification task: the way in which a text is written can basically be a digital fingerprint of the author. It is indeed known that fiction and non-fiction texts present different writing strategies: indicators of who has written the text can be seen in verbal morphology, distribution of subordinates, depth of the parse tree and various others.
Sociolinguistics studies like [17] have shown that people use grammatical features to signal the speakers' membership in a demographic group, with a focus on gender. One known measure is the use of pronouns: male authors, for example, use more plural pronouns (we, us, they, them) in fiction and more male third-person pronouns (he, him) in both fiction and non-fiction.
Going deeper with these analyses, and trying also using different tokenizers, POS tagger and treebanks, maybe specific ones based on fanfictions' language, could surely improve the results.
Another interesting feature we think could be very interesting to explore is the role of Artificially Intelligent (AI) authors: their linguistic behaviour changes constantly as these technologies and Language Models used are still in the process of developing (e.g. GPT2, GPT3, general Chatbots) and it has not yet been profiled in any way (one of the very few examples of a similar analysis is [2]).
As future work, we think that a mix of the Writeprints features with the embedding of the text could be an interesting approach to investigate. Another thing to investigate in the future could be features related to the specific fandom (topic) to which the texts belong to, as suggested from other works.

REFERENCES

[1] Ahmed Abbasi and Hsiu-chin Chen. 2008. Writeprints: A Stylometric Approach to Identity-level Identification and Similarity Detection in Cyberspace. *ACM Transactions on Information Systems* 26 (03 2008), 1–29. https://doi.org/10.1145/1344411.1344413

[2] Nawaf Ali, Musa Hindi, and Roman Yampolskiy. 2011. Evaluation of authorship attribution software on a Chat bot corpus. *Proc. 23rd Int. Symp. Information, Communication Automation Technologies* (10 2011). https://doi.org/10.1109/ICAT.2011.6102123

[3] Emir Araujo-Pino, Helena Gómez-Adorno, and Gibran Fuentes-Pineda. 2020. Siamese network applied to authorship verification. *Working Notes of CLEF* (2020).

[4] Shlomo Argamon, Jonathan Fine, and Anat Shimoni. 2003. Gender, Genre, and Writing Style in Formal Written Texts. *Text* 23 (12 2003). https://doi.org/10.1515/text.2003.014

[5] Janek Bevendorff, Bilal Ghanem, Anastasia Giachanou, Mike Kestemont, Enrique Manjavacas, Martin Potthast, Francisco Rangel, Paolo Rosso, Günther Specht, Efstathios Stamatatos, Benno Stein, Matti Wiegmann, and Eva Zangerle. 2020. Shared Tasks on Authorship Analysis at PAN 2020. In *Advances in Information Retrieval*, Joemon M. Jose, Emine Yilmaz, João Magalhães, Pablo Castells, Nicola Ferro, Mário J. Silva, and Flávio Martins (Eds.). Springer International Publishing, Cham, 508–516.

[6] Janek Bevendorff, Benno Stein, Matthias Hagen, and Martin Potthast. 2019. Generalizing Unmasking for Short Texts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 654–659. https://doi.org/10.18653/v1/N19-1068

[7] Sebastian Bischoff, Niklas Deckers, Marcel Schliebs, Ben Thies, Matthias Hagen, Efstathios Stamatatos, Benno Stein, and Martin Potthast. 2020. The Importance of Suppressing Domain Style in Authorship Analysis. *CoRR* abs/2005.14714 (May 2020). https://arxiv.org/abs/2005.14714

[8] Benedikt Boenninghoff, Julian Rupp, Robert M Nickel, and Dorothea Kolossa. 2020. Deep bayes factor scoring for authorship verification. *arXiv preprint arXiv:2008.10105* (2020).

[9] Eric Brill. 1992. A Simple Rule-Based Part of Speech Tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing* (Trento, Italy) *(ANLC '92)*. Association for Computational Linguistics, USA, 152–155. https://doi.org/10.3115/974499.974526

[10] Łukasz Gagała. 2020. Authorship verification with prediction by partial matching and context-free grammar. *Working Notes of CLEF* (2020).

[11] Oren Halvani, Lukas Graner, and Roey Regev. 2020. Cross-domain authorship verification based on topic agnostic features. *Working Notes of CLEF* (2020).

[12] Catherine Ikae. 2020. UniNE at PAN-CLEF 2020: Author verification. *Working Notes of CLEF* (2020).

[13] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. arXiv:1909.10351 [cs.CL]

[14] Mike Kestemont, Enrique Manjavacas, Ilia Markov, Janek Bevendorff, Matti Wiegmann, Efstathios Stamatatos, Martin Potthast, and Benno Stein. 2020. Overview of the Cross-Domain Authorship Verification Task at PAN 2020. In *CLEF 2020 Labs and Workshops, Notebook Papers*, Linda Cappellato, Carsten Eickhoff, Nicola Ferro, and Aurélie Névéol (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-2696/

[15] Mike Kestemont, Michael Tschuggnall, Efstathios Stamatatos, Walter Daelemans, Günther Specht, Benno Stein, and Martin Potthast. 2018. Overview of the Author Identification Task at PAN-2018: Cross-domain Authorship Attribution and Style Change Detection. In *Working Notes of CLEF 2018 - Conference and Labs of the Evaluation Forum, Avignon, France, September 10-14, 2018 (CEUR Workshop Proceedings, Vol. 2125)*, Linda Cappellato, Nicola Ferro, Jian-Yun Nie, and Laure Soulier (Eds.). CEUR-WS.org. http://ceur-ws.org/Vol-2125/invited_paper_2.pdf

[16] Alon Kipnis. 2020. Higher criticism as an unsupervised authorship discriminator. *Working Notes of CLEF* (2020).

[17] Britta Mondorf. 2002. Gender Differences in English Syntax. *Journal of English Linguistics* 30, 2 (2002), 158–180. https://doi.org/10.1177/007242030002005 arXiv:https://doi.org/10.1177/007242030002005

[18] Anselmo Peñas and Alvaro Rodrigo. 2011. A Simple Measure to Assess Non-response.

[19] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084 [cs.CL]

[20] Stuart Russell and Peter Norvig. 2002. Artificial intelligence: a modern approach. (2002).

[21] Ahmad Shehabat, Teodor Mitew, and Yahia Alzoubi. 2017. Encrypted Jihad: Investigating the Role of Telegram App in Lone Wolf Attacks in the West. *Journal of Strategic Security* 10, 3 (2017), 27–53. http://www.jstor.org/stable/26466833

[22] Janith Weerasinghe and Rachel Greenstadt. 2020. Feature Vector Difference based Neural Network and Logistic Regression Models for Authorship Verification. In *Working Notes of CLEF 2020-Conference and Labs of the Evaluation Forum, Thessaloniki, Greece, September 22-25, 2020*. CEUR-WS. org.