Novel Approaches to Authorship Attribution

Submitted in partial fulfilment
of the requirements of the degree of

Master of Arts

of University of Groningen

Gareth Terence Bryant Dwyer

Groningen, the Netherlands
July 2017

Abstract

This is pretty abstract.

ACM Computing Classification System Classification

Thesis classification under the ACM Computing Classification System (1998 version, valid through 2013) [16]:
I.2.7 [Natural Language Processing]: Web Corpora

General-Terms: Corpus, Corpora, Natural Language, Concordancer

Acknowledgements

I would like to thank all the people who gave me money and my supervisor too. Also some other people.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Afroz *et al.* (2014) show how authorship attribution can be used for deanonymizing criminals in underground internet forums.

Chapter 2

Background

In this chapter, we will review prior literature relating to authorship attribution. We will further review prior literature that relates to the related fields of text classification and language modelling. Note that many of the articles that we make reference to are arXiv[1] preprints and not all of them have been published in peer-reviewed journals. Due to the fast-moving nature of the field, we believe that this work should be taken as part of the prior literature even without having proven itself through peer-review.

Stamatatos (2009) has already created a comprehensive survey, summarizing and comparing various techniques that have been used for authorship attribution, including feature engineering and classification methods. In order to avoid repeating Stamatatos's work, we therefore focus here on newer research (published after his survey), and research that is closely connected to our own (that is, work which uses similar features, classification techniques, and datasets).

2.1 Authorship Attribution

In this section we present a review of prior work that is closely related to our research on Authorship Attribution (AA). The two main sub-tasks of AA, as discussed before, are Authorship Identification (AID) and Authorship Verification (AV). We discuss each of these in turn. By some definitions, related tasks such as age profiling, gender profiling, personality profiling, and native language identification, also fall under the AA umbrella.

[1] https://arxiv.org/

We discuss these tasks briefly later, but for our work we will regard AA as consisting of AID and AV.

2.1.1 Authorship Identification

Authorship Identification (AID) is the most well-known Authorship Attribution task. For the AID task, the goal is to predict which of a closed set of candidate authors is the author of an *unknown* text, where the unknown text is of disputed authorship. (Stamatatos, 2009) describes this task as follows:

> In the typical authorship attribution problem, a text of unknown authorship is assigned to one candidate author, given a set of candidate authors for whom text samples of undisputed authorship are available. From a machine learning point-of-view, this can be viewed as a multi-class single-label text categorization task.

Many approaches to authorship attribution have been attempted over the last three decades, including supervised and unsupervised approaches. For supervised approaches, the features used have varied greatly. Stamatatos (2009) lists 20 commonly used features, which he breaks down into the categories of *lexical* (for example, word n-grams), *character* (for example, character n-grams), *syntactic* (for example, parts of speech tags), *semantic* (for example, synonyms), and *application specific* (for example, language specific dictionary).

Intuitively, when attempting to identify the author of a text, it seems that very unusual words and phrases used by that author would be helpful. In practice, however, the unique way that an author uses very common words and phrases is far more useful. Looking only at very common function words (for example, 'the', 'and', 'he', etc), can in many cases be enough to reliably distinguish one author from another. (Kestemont, 2014) talks about why function words are important for authorship attribution. He gives four main points regarding function words, namely that all people who write in the same language will use the same function words; that function words appear with high frequency in almost all texts; that function words are not effected by the *topic* of the text; and that function words seem"eems less under an authors conscious control".

(Koppel *et al.*, 2009) also support the idea that function words are good for Authorship Attribution. They state:

> The reason for using FWs [function words] in preference to others is that we do not expect their frequencies to vary greatly with the topic of the text, and hence, we may hope to recognize texts by the same author on different topics. It also is unlikely that the frequency of FW use can be consciously controlled, so one may hope that use of FWs for attribution will minimize the risk of being deceived

However, Kestemont (2014) also argues that function words are more useful in English settings than for other languages, because English, as a language that does not make heavy use of inflections, relies more on function words than other languages do. He believes that character n-grams can often provide an adequate representation of function words in a text, while also being "sensitive to the internal morphemic structure of words". That is, it is likely that models which rely on character n-grams as features will be more language independent than word-based models, as character n-grams can capture distinctive function word usage, while also capturing distinctive inflection usage.

Feature engineering is not always desirable as it requires a topic specialist to construct task-specific features. It can also make classification tasks less efficient (for example, if a parts-of-speech tagger is needed to extract a specific feature then classification performance will be drastically reduced), and less generalizable (a model that relies on parts-of-speech tag is less likely to perform well cross-lingually, especially for languages which do not have good taggers). Furthermore, recent research has shown that manual feature engineering can result in worse accuracy than in cases where the classifier is relied upon to infer features from rawer input. (FIXME cite n-GrAM)

Recently, authorship attribution models which rely on almost no manual feature engineering have proven to perform well. For example, Bagnall (2015) achieved the top score at an Authorship Attribution shared task, using only the characters of each text as features. We discuss his approach in more detail in Section FIXME.

With the above in mind, we focus on approaches that do not require heavy feature engineering, and which are therefore more applicable across different domains and languages.

Much of the prior work related to AID uses settings that are not common in real-world authorship attribution tasks. Luyckx (2011) talks about the scalability issues relating to authorship identification and criticises prior research on two main points. First, earlier work often uses a very small number of candidate authors for AI tasks (sometimes only two). Second, this work often uses very long texts (often each text is a full-length book). By contrast, in practical AA tasks there is often only a short fragment of text available, and a large number of candidate authors. Luyckx states:

> authorship attribution 'in the wild' may entail thousands of candidate authors with often small sets of data or only very short texts, in substantially more topics, genres, and registers

It is therefore desirable to experiment with AID tasks that are closer to the settings found "in the wild". That is, we want our methods to perform well in cases where their are many candidate authors and limited text available for each. More modern papers often acknowledge and address the issues raised by Luyckx, but it is not unusual to still see studies that ignore these issues. For example, Akimushkin *et al.* (2017) provide an extensive analysis on using text networks for AA, but present results on a dataset consisting of only eight authors, and using ten full-length books for each author.

Abbasi & Chen (2008) present research that has closer ties to practical AA problems. They focus on attempting to identify authors in "cyberspace", and use datasets consisting of emails and online forums. They use the Enron dataset (Klimt & Yang, 2004) which we also use, and describe in Section 4. Our work contrasts with theirs in that they use many of the features described above, while in most of our models, we attempt to solve the task using as few features as possible.

Somers & Tweedie (2003) look at some basic stylometry features such as lexical richness and take special note of Alice Through the Needle's Eye, sometimes claimed to be the best Pistache. REMOVE.

2.1.2 Authorship Verification

Authorship Verification (AV) is an AA task in which the goal is to predict whether two texts are written by the same author. It is sometimes formulated as deciding whether or not a *specific* author wrote a disputed text. Therefore, it can be thought of in two ways. First, it can be modelled as a one-class classification problem, in which we attempt to distinguish a single class (or a single author) from all other possible texts.

Koppel & Schler (2004) takes this approach and in explaining why AV is a more interesting and realistic task than AID, he states:

> If, for example, all we wished to do is to determine if a text was written by Shakespeare or Marlowe, it would be sufficient to use their respective known writings, to construct a model distinguishing them, and to test the unknown text against

> the model. If, on the other hand, we need to determine if a text was written by Shakespeare or not, it is very difficult if not impossible to assemble an exhaustive, or even representative, sample of not-Shakespeare.

To keep the advantages of AV over AID, but to also solve the problem of representing "not-Shakespeare", (Koppel *et al.*, 2009) later argued that it might be better to think of AV as a two-class problem. He states:

> Verification can be thought of as a one-class classification problem (Manevitz & Yousef, 2001; Scholkopf, Platt, Shawe- Taylor, Smola, & Williamson, 2001; Tax, 2001). But perhaps a better way to think about authorship verification is that we are given two example sets and are asked whether these sets were generated by the same process (i.e., author) or by two different processes.

Koppel has pushed the idea of AV being a more important and interesting task than AID more strongly over the years. He has stated that "a solution to [authorship verification] can serve as a building block for solving almost any conceivable authorship attribution problem" (Koppel *et al.*, 2012b) and has called it the "fundamental problem of Authorship Attribution" (Koppel *et al.*, 2012b,a).

Luyckx & Daelemans (2008) are also proponents of Authorship Verification. They state:

> Most studies also use sizes of training data that are unrealistic for situations in which stylometry is applied (e.g., forensics), and thereby overestimate the accuracy of their approach in these situations. A more realistic interpretation of the task is as an authorship verification problem[...].

However not everyone uses the same definitions or terminology for Authorship Verification tasks. Another way to distinguish between AV and AID is by distinguishing between 'closed-world' and 'open-world' tasks. In AID, the world is 'closed' because we can enumerate all the potential authors for a given text. In AV, the world is 'open', because either the two texts are written by the same author, or they are not. In the latter case, we do not attempt to define who is the actual author of the unknown text. This distinction (and terminology) is discussed by Stolerman *et al.* (2011) who aim to reconcile theoretical AA work with practical issues. They present research on "Breaking the Closed-World Assumption in Stylometric Authorship Attribution", and argue that much theoretical research is impractical as it assumes a closed-world setting.

They introduce the *classify-verify* method for AA tasks, which adds a second step to a traditional classification approach in which the classifier is taught to "abstain" in certain cases, and argue that this is a good compromise between closed- and open-world tasks.

Stolerman (2015) presents a comprehensive review of Authorship Verification methods. He distinguishes between two classes of authorship tasks, namely *one-class* problems and textittwo-class problems. The former refers to tasks in which we attempt to distinguish a single class (for example, a single author) against all other classes (for example, all other possible authors). Two-class problems refer to tasks in which we attempt to assign one of two labels (for example, *same-author* or *different-author* to each instance. Stolerman notes that there is no need to discuss the more general n-class classification as any n-class problem can be reduced to multiple one-class or two-class tasks.

Stolerman [PhD thesis on Authorship Verification] discusses at length the need for authorship attribution research to focus more on practical tasks, and to move away from the traditional closed-task setups that have dominated previous work. He states

> The standard closed-world authorship attribution domain, however, is abundant with datasets that can be trivially formulated to test verification. If more datasets are to be used and tested, it can assert the usability of current and future verification methodologies, with emphasis on which techniques are suited to what problems. Verification methods should be tested on datasets that challenge with a high number of potential authors, taking pure one-sided learning approaches, limited amounts of training data, texts with real-world characteristics and the like.

Central to the verification task is the idea of similarity. Because we have two texts, a known and an unknown, we want to be able to tell how stylistically similar these two texts are. This has led researches to propose many different ways of representing similarity, both in terms of features used as well as custom distance measures. Halvani *et al.* (2016) propose a distance function which is a modified version of Manhattan Distance, and show a novel similarity-based authorship verification that generalises well over different genres and languages. Halvani *et al.*'s work is mainly of interest to us because they show results for many different PAN datasets, which we also use.

A less usual approach to modelling similarity is by using compression models. Data compression exploits patterns in text to efficiently reduce the storage space required to store a representation of that text. If a compression model 'trained' on a *known* text is also able to efficiently compress an *unknown* text, then at least some patterns in the

known text are also present in the unknown text. (Stamatatos, 2009) discusses some older work that uses compression models for AA, and more recently (Halvani *et al.*, 2017) has investigated these methods again. Although we do not use compression models in our current work, this research is related to ours because character-based neural language models, which we do use and describe in detail in Section **??**, rely on a similar idea: that a system trained to find low-level patterns in a known text can be used to evaluate an unknown text.

Luyckx & Daelemans (2008) describe why the verification task is more interesting than the identification one. They use a high-school essay corpus of 145 authors. (Koppel & Winter, 2014) shows how to use the Imposter's algorithm for verification. Shows MiniMax similarity.

(Halvani *et al.*, 2016) present results for authorship verification on a several corpora, including PAN. They achieve a median accuracy of 0.7.

[@halvani2017authorship] evaluate simple and fast compression-based models for authorship verification. They compare their system to Bagnall and GLAD, and use the PAN dataset.

2.1.3 Stylometry and writing style

We have discussed how it is possible to attribute a work to a specific author based on specific stylistic features. However, it is important to note that the concept of authorship style is not well defined nor well understood, and different studies often refer to very different concepts under the same label of "writing style". Gatt & Krahmer (2017), in an extensive survey on text generation, state:

> What does the term 'linguistic style' refer to? Most work on what we shall refer to as 'stylistic nlg' shies away from a rigorous definition, preferring to operationalise the notion in the terms most relevant to the problem at hand.

For authorship attribution, style is closely associated with, for example, which parts-of-speech tags authors choose to use (a specific author might use a lot of adjectives), how they choose to punctuate (some authors are proud of never having used a semi-colon in their lives; others use them frequently), or how long they typically make their sentences.

By contrast, "style" often refers to a more general concept which includes the register, formality, or tone of a specific text. For example, (Jhamtani *et al.*, 2017) show how it is possible to automatically "translate" between writing styles by taking modern text as input and producing text in the style of Shakespeare as output, while keeping the meaning of the text the same. In order to achieve this, we need to be able to discriminate between *content* and *style*. If we could do this reliably, authorship attribution would be a much easier task, as we could simply compare various writing styles needing to deal with content. However, (Jhamtani *et al.*, 2017) needed parallel corpora to achieve the results that they did. These were available as all of Shakespeare's works have been translated into "modern" English manually. There is ongoing work on automatic style transfer. Kabbara & Cheung (2016) propose the opposite of (Jhamtani *et al.*, 2017), that is, they present a proposal in which they aim to build a Recurrent Neural Network system that, trained on Shakespeare and Simple-English Wikipedia, could translate works written in the style of Shakespeare into a more modern and easier to read style.

While we have two different versions of Shakespeare's work (the original version and the modernized version), this is uncommon. In nearly all cases, when we examine text written in different styles, the content differs as well. One exception is the book *Exercises in Style* (Queneau, 1981), in which the same short story is re-written 99 times in different styles. The book was written in French, but has since been translated into over 30 other languages. While this work is of interest to any studies that relate to writing style, the book is too short to be of use in teaching machine learning models to discriminate between styles more generally, and moreover although the author consciously varies his style for the 99 variations, they are all still written by the same author (or translator). Therefore much of the subconscious writing style variations (such as the use of function words discussed above) would be lost.

2.1.4 Related tasks

By some definitions, other author profiling tasks also form part of general AA. These include, Age Profiling, Gender Profiling, Personality Profiling, and Native Language Identification. For age profiling the task is usually to predict which of several age ranges an author belongs to, looking only at text written by that author. For gender profiling, we attempt to discriminate between male and female authors. Personality profiling is less common, and usually attempts to predict Myers-Briggs personality types (Myers, 1962). Native Language Identification is the task of predicting the authors first language from

text produced in a second language (usually English). This has practical links to forensic linguistics, in that even if we cannot identify the exact author of an unknown text, it is often still useful to identify their nationality by using their native language as a proxy, and it's ties to Authorship Attribution are discussed by (Stolerman, 2015).

We do not investigate these tasks closely in the current work, but studies that focus on these areas is related to ours in that the systems that work well for these tasks also work for AID and AV, due to the fact that the tasks all fall under a general assumption of classifying authors based on their writing style. Specifically, the annual PAN AA shared task, which is one of our primary sources for AID and AV prior work, runs parallel shared task on Authorship Profiling, with a focus on age and gender tasks. Methods which use Support Vector Machine classifiers have shown to outperform other methods in all of these tasks. Similarly (Verhoeven *et al.*, 2016) use SVMs and TF-IDF vectors for gender and personality profiling.

Detecting the language style of non native speakers https://arxiv.org/pdf/1704.07441.pdf [@rudzewitz2016exploring] link Authorship Attribution with Plagiarism Detection (and Short Answer Assessment).

2.2 General NLP and text classification

Support Vector Machines

> Comparative studies on machine learning methods for topic-based text categorization problems (Dumais et al. 1998; Yang 1999) have shown that in general, support vector machine (SVM) learning is at least as good for text categorization as any other learning method and the same has been found for authorship attribution (Abbasi & Chen 2005; Zheng et al. 2006).

(Koppel *et al.*, 2009)

2.2.1 Neural networks for authorship attribution

As previously discussed, there has been very little work in using Neural Networks for Authorship Attribution. This is partly because in Authorship Attribution tasks, as discussed above, there is often very limited training data available. Neural Networks have proven

to be very powerful in many text classification tasks, but they often rely on huge amounts of training data to achieve good results. Here, we discuss some existing attempts that use neural approaches to Authorship Attribution.

Bagnall (2015, 2016) has achieved the top rank at recent PAN Authorship Attribution shared tasks. He competed in and won two shared tasks over two years. The first task was an AV task, while the second was an "authorship clustering" task, which is a more complicated task but which can still be remodelled as AV. Bagnall's approach is interesting for three reasons. First, he uses a neural network approach which outperformed competitors SVM models. Second, he uses very little manual feature engineering, relying only on character sequences. Third, instead of training a discriminative classifier to discriminate between authors, he trains separate generative models for each candidate author. He then sees which author-model best fits an unknown text. To convert this into a verification task, he sees

(Bagnall, 2015)

Work directly related to ours... [@shrestha2017convolutional] use CNNs for authorship attribution on short texts (tweets). They cite Bagnall and PAN, and discuss what the CNN learns.

[@yogatama2017generative] discuss and compare generative vs discrimative LSTM models for classification tasks.

[@weissenborn2016neural] Show how most NLP tasks rely on two sequences of text: either sequence to sequence, as in MT, or dual-sequence for many classification tasks.

[@deng2015deep] compare generative and discriminative models, saying that generative models can be better for discriminative tasks when training data is limited as they can converge faster than generative models

[@chrupala2013text] are the first(?) to use character embeddings. They use them to recognise code segments within text.

2.2.2 Neural networks

[@raghu2016expressive] talk about the relation between the structure of a neural network and the functions that it is able to compute. They state that lower layers are more

important and are more sensitive to noise and optimizations, while higher layers can model exponentially more complex functions.

[@spieckermann2015multi] does an extensive investigation of multi-task and transfer learning using RNNs. Not much specific to language modelling.

[@tiflin2012lstm] Show that LSTMs can be used for signature verification (SatNac, UWC paper).

2.2.3 Siamese Neural Networks

Siamese neural networks are a customized neural network architecture in which two sub-networks share weights which are simultaneously updated during training. A joining neuron learns a distance function between the two networks, and outputs a single similarity measure. Thus the network as a whole learns a custom distance function between pairs of inputs. Siamese neural networks are designed for verification tasks, in which we want to classify the relation between pairs of inputs, and it therefore in theory fits the AV task very well. They have been used mainly for image verification tasks, as discussed below, but they are now gaining in popularity for text classification tasks such as question answering.

Here we provide a general introduction and review of Siamese Neural Networks, followed by a summary of work which uses these networks for text classification tasks.

Siamese Neural Networks and Image Verification

The concept of a Siamese Neural Network was first introduced by Bromley *et al.* (1993), who used it for Signature Verification. With signature verification, the task is to predict whether two signatures are signed by the same hand, or if one of the them is a forgery. The authors compare the two identical sub-networks, or "legs", of the siamese network to feature extraction, as these learn which features are important, while the joining neuron measures the similarity between the two legs. If the measured distance between a known signature and an unknown one is greater than some threshold, then the unknown signature is rejected as being a forgery.

Since then, Siamese Networks have been used for similar verification tasks, such as face verification. This is similar to signature verification, but it is used, for example, for access

control. A new picture of a person is taken and compared to one that is stored on record (for example, in an ID card or passport). If the system predicts that the new picture is the same as the stored one, the person is 'verified' and allowed access. (Chopra *et al.*, 2005) describe how a siamese neural network can be used to solve this task as follows:

> We present a method for training a similarity metric from data. The method can be used for recognition or verification applications where the number of categories is very large and not known during training, and where the number of training samples for a single category is very small. The idea is to learn a function that maps input patterns into a target space such that the L_2 norm in the target space approximates the semantic distance in the input space. The method is applied to a face verification task. The learning process minimizes a discriminative loss function that drives the similarity metric to be small for pairs of faces from the same person, and large for pairs from different persons.

Similarly Zhu *et al.* (2017) achieved good results using siamese neural networks for face verification (called person reidentification in their work). A more complicated extension of image verification is automatic scene detection for films. Siamese networks have shown promising results for this task as well by doing multiple frame-wise verification sub-tasks, deciding which frames belong to the same scene, and which to different scenes in order to detect when the scene changes (Baraldi *et al.*, 2015).

The face verification task, in which real-world settings often mean that ta large or unknown number of classes needs to be taken into account and the training data per class is highly limited, is highly similar to the AV task that we already described. Although images are used for the face verification task, while we need to look at text for the AV task, because neural networks have shown to be proficient at classifying both images and text, and because the siamese architecture has been shown to be successful for the face verification task, it is likely that a similar architecture would work well for the AV task.

The task ex

[@naaman2017learning] use a siamese (RNN) network for learning a pronunciation similarity function. Maybe I should feed the predictions from generic and fine-tuned generative models into siamese network?

[@liu2013probabilistic] a master's thesis on Siamese networks, mainly wrt. images.

The point is as zxcv you can see

One-Shot learning is a way to use Neural Networks even when there is only a small amount of training data available. Nice intro: https://sorenbouma.github.io/blog/oneshot/ (using siamese networks); paper http://www.cs.cmu.edu/ rsalakhu/papers/oneshot1.pdf; for immitation learning (https://arxiv.org/abs/1703.07326)

[@baraldi2015deep] create a Siamese Network for learning scene detection in videos. Deciding if two frames are from the same scene is related to deciding if two texts are written by the same author. Interestingly, they first teach the network to be able to identify objects, such as a tree, by using Image Net, and then use the Siamese network to figure out which features are indicative of a shared scene. This is simlar to first training a generic language model, which learns language features, and then using a siamese network to work out which features are indicative of the same author.

[@hosseini2015similarity] use a Siamese network to help with OCR – instead of doing full analysis of handwritten text in an image, they can find similar images that have already been OCRd.

2.2.4 Siamese networks for NLP

Neculoiu *et al.* (2016) use a Siamese Network to match job titles that refer to the same (or similar) positions, but which are lexically different. For example, in the technology industry, "software architectural technician Java/J2EE" may describe the same job as "Java Engineer".

[@yin2015abcnn] use three models for Sentence Pairs, including (Attention Based) Siamese Networks

[@mueller2016siamese] use siamese networks to achieve a new baseline in a sentence similarity task.

[@hoffer2015deep] created a so-called "Triple Network", which extends the idea of a Siamese Network. It takes three inputs – X, X+ and X-, where X and X+ are the same class and X- is a different class.i

2.2.5 Transfer Learning and Style Transfer

[@shen2017style] use 'style' transfer where style refers to sentiment or ciphertext. They use GaNs and VAEs. Nice discussion of previous style transfer works.

[@zoph2016transfer] look at using Transfer Learning to solve data scarcity issues

[@shin2016generative] use a 'teacher-learner' transfer model, in which the student is trained on the output of the teacher(?).

[@riemer2017representation] IBM paper on multi-task learning and transfer learning for text classification (Twitter sentiment). "The very popular strategy of fine-tuning a neural network involves first training a neural network on a source task and then using the model to simply initialize the weights of a target task network up to the highest allowable common representation layer"

[@lalor2017improving] talk about the fine-tuning a neural model by adding supplemental data. They distinguish this from transfer learning, where transfer learning uses data from a different task, while fine-tuning uses more data from the same task.

[@mcgraw2016personalized] use an LSTM for speech recognition on mobile devices, such as "Call Jacob". They personalise language models by adding the users' contacts (bias the language models on the fly). This can be seen as a crude version of transfer learning?

https://arxiv.org/pdf/1707.01161.pdf Use seq-to-seq models to translate modern english into shakespeare style english. they pretrain embeddings based on dictionaries.

2.2.6 Language Modelling

Yoav's new paper on stylistic NLG https://arxiv.org/pdf/1707.02633.pdf

Why is character-level language modelling so effective for Authorship Attribution? The two most helpful features provided by character level modelling are *affixes* and *punctuation*, both of which are better captured by character-level models than word-based models. [@sapkota2015not]

@gatt2017survey present a comprehensive modern survey (111 pages) on Natural Language Generation. They include a section on generating language in a specific genre or author style, and conclude that there is much work still to be done in this area.

[@karpathy2015unreasonable2] shows how well an RNN can be used to model different styles of language using only characters.

[@mikolov2012subword] look at using 'subwords' instead of characters. They discuss the tradeoffs between word and character level language modelling.

[@audhkhasi2017end] use a CNN to train character embeddings and then feed these embeddings into a RNN to create a language model. They do this as part of an end-to-end "ASR-free" keyword search from speech system.

[@goga2013exploiting] show that language modelling is hard – to deanonymize users on Yelp and Twitter they found that language was the least useful feature, with location and timestamps of updates being much better at identifying authors.

[@sutskever2011generating] Discuss character level RNNs in detail and construct a large network (trained for five days on 8 GPUs).

Chapter 3

Method

As previously discussed, there are a number of different ways to appraoch authorship attribution tasks. Specifically, we experiment with both authorhip identification and authorship verification tasks, using different methods. These methods can be broadly broken into

- Statistical methods, in which we compare texts using (unsupervised) statistical methods.
- Support Vector Machine (SVM) methods, in which we use SVMs to discriminate between authorship styles.
- Neural Network approaches, in which we experiment with various formulations of AA tasks and various models, including Siamese Networks and generative language modelling.

In the rest of this chapter, we explain the set up for various experiments covering these three methods. Due to practical and theoretical limitations, we did not experiment with all methods on all tasks. Nor did we use every dataset for every experiment. In !REF, we present an overview of which methods we tried on which datasets.

3.1 Unsupervised statistical approaches

Previously, we discussed the distinction between *intrinsic* and *extrinsic* approaches to authorship attribution. Recall that the former relies only on comparing the target texts

(e.g. a known and an unknown text) to each other, while the latter compares the target texts to a corpus of external texts.

In this section, we describe our experiments involving *extrinsic* approaches. Supervised machine learning has become the go-to method for many text classification tasks, but the resulting models are often seen as a "black box". Data goes in and predictions come out, and it can be difficult to justify or explain these predictions. Here, we explain how we can meaningfully analyse texts using some basic statistical approaches, including correlation techniques, on a number of different features. For these techniques, we experimented only with authorship verification tasks.

3.1.1 Authorship verification tasks

To decide if two texts are written by the same author, we can compare various features in each text to some large corpus of (preferably similar) texts. If, for example, both texts have (when compared to the corpus) a higher-than-average sentence length, or a lower-than-average adjective frequency, then we have some small piece of evidence that the two texts were written by the same author. If the two texts relate to the corpus similarly over a variety of different features, then we have more evidence that the two texts share an author.

We experiment with two ways of comparing our target texts to a corpus. Firstly, we use a simple count-based approach based on supporting and opposing points for the hypothesis that the texts share an author. Secondly we use a correlation analysis, to see if the two texts diverge from the corpus in a similar fashion.

We use lexical and syntactic features for these experiments, including sentence length, word length, various readability scores, parts-of-speech (PoS) tag relative frequency, and frequency of the most-common function words. Because we use more features here than in our later experiments, we carry out these experiments only on the smaller English datasets, C50 and the English parts of PAN.

Count-based statistical experiments First, we use a simple count-based approach. If two texts, compared to the corpus, either:

a) Both have a feature that is higher-than-average (e.g. both texts have a higher-than-average adjective frequency), or

b) Both have a feature that is lower-than-average (e.g. both texts have an average sentence length that is lower than the average of the corpus)

then we take this to be a point supporting the hypothesis that the texts share an author.

If there is a feature such that one text is higher-than-average while the other is lower-than-average (e.g. Text 1 uses more commas than the corpus-average while Text 2 uses fewer commas than the corpus-average), then we take this to be an opposing point.

We then classify texts into same-author or different-author classes based on whether there are more supporting points than opposing ones.

Correlation based statistical experiments A more refined way is to look do a correlation test between the two sets of scores. If one text has a much higher than average sentence length, and the other has only a slightly higher than average sentence length, then this should be less indicative of same-author than if the two texts both have a much higher than average sentence length.

If two texts arewritten by the same author, we expect the exact way that each diverges from the reference corpus to correlate quite strongly. If the texts are written by different authors, we expect a weaker correlation.

3.2 Support vector machine approaches

Support Vector Machines (SVMs) have proven to be efficient and powerful for a wide-variety of text-classifciation tasks. For all of our multi-class SVM experiments, we use a one-against-all strategy. That is, if we have 50 candidate authors, we train 50 SVMs, and each one is trained to discriminate in a binary fashion between its author and all other authors.

Because SVMs are fast to train and scale well to large datasets, we present SVM-based experiments for all of our datasets.

3.2.1 Authorship identification tasks

For the identification task, we can simply train SVMs on the authors knowns texts, and attempt to predict the author of unknown texts from the pool of candidate authors. While this is perhaps the most-common set up for authorship attribution tasks, it is also the least interesting. Generally, good results are only feasible for a small pool (fewer than 100) candidate authors, and practically there are very few cases where we want to know which of small, well-defined set of authors wrote a particular text. Nonetheless, we use this set up to experiment with different numbers of authors and different features, predominately word and character n-grams represented as Term Frequency - Inverse Document Frequency (TF-IDF) vectors.

3.2.2 Authorship verification tasks

Normally for text classification tasks, the goal is to apply a label to each text. For authorship verification tasks, we need to say something meaningful about the relation between a pair of texts. A naive approach might be to concatenate the two texts and then attempt to assign a "yes" or "no" label. However, we don't want to learn rules such as "if either text contains the word *discombobulation* then the two texts are likely to be written by the same author" which is unfortunately the kind of rule that a traditional classifier would learn if we trained it on concatenated pairs of texts.

Therefore, to learn something meaningful about text pairs, we need to look at the similarity between the two texts. Often, the cosine distance between the vector representations of each text is used for such tasks. However, two different authors often write texts that have a strong cosine similarity (for example, two authors writing about the same topic), and conversely a single author will often produce two texts that appear very dissimilar by the cosine metric (for example, if the author writes on two different topics).

By subtracting TF-IDF vectors of each text, we get a feature set that ismore meaningful. The SVM can learn which word- and character n-grams are indicate authorship, and can learn rules such as "if the count of the word "the" is similar in both texts, then it is likely that they are written by the same author".

3.3 Neural network approaches

Neural Networks (NNs), especially Recurrent Neural Networks (RNNs) with Long Short-Term Memory (RNN-LSTMs) or with Gated Recurrent Units (RNN-GRUs) have been the poster-child of many Natural Language Processing (NLP) advancements in the last few years. As discussed before, they have failed so far to become state-of-the-art for Authorship Attribution tasks. This is for several reasons, including:

- RNNs are difficult to train and not yet fully understood
- RNNs usually require much larger amounts of data than existing AA datasets
- The feedback cycle for RNN classifiers can be much longer than alternative approaches such as SVMs. For example, for one of our experiments FIXME REF, the SVM took about three minutes to train and evaluate, while the RNN model took over 14 hours.

Our experiments with Neural Network classification can be broken into three main categories:

- **Discriminative Text Classification**: Similarly to our SVM identification experiments, we model the authorship identification problem as a traditional multi-class classification task. Instead of TF-IDF vectors, we use word embeddings, and instead of the SVM classifier, we use a neural network model.
- **Generative, author-specific language modelling**: Following recent advances in neural character-based language modelling, it is possible to generate text in the style of a specific author (for example, Karpathy FIXME REF). By training language models to mimic the writing style of specific authors, we can classify unknown texts by seeing which author-specific language model best models that text.
- **Siamese neural networks**: We use a neural network that contains two sets of identical weights and learns a custom distance function between pairs of inputs. This fits our verification task very well.

Authorship identificationtasks

Discriminative classifiers

TODO or remove

Generative Language Modelling

If we can generate text in the style of a specific author, we can also recognise it. Previous work has shown that generative classifiers can be more effective than discriminative ones, even for discriminative tasks. For these experiments, we analyse various ways to create personalized language models, including different architectures, and using transfer learning.

3.3.1 Authorship verification tasks

Discriminative classifiers

Generative language modelling

Because identification tasks can often be reformulated as verification tasks, and vice-versa, we also attempt some of the verification tasks using the generative personalized models described above. For verification, we train personalized language models for each *known* text. Then we compute the cross entropy for each unknown text, for each model. If the unknown text has a lower cross entropy than more than half of the personalized models, that provides an indication that the two texts are by the same author.

Siamese networks

TODO

Chapter 4

Data

Chapter 5

Models

Chapter 6

Results

Chapter 7

Conclusion

References

Abbasi, Ahmed, & Chen, Hsinchun. 2008. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Transactions on Information Systems (TOIS)*, **26**(2), 7.

Afroz, Sadia, Islam, Aylin Caliskan, Stolerman, Ariel, Greenstadt, Rachel, & McCoy, Damon. 2014. Doppelgänger finder: Taking stylometry to the underground. *Pages 212–226 of: Security and Privacy (SP), 2014 IEEE Symposium on*. IEEE.

Akimushkin, Camilo, Amancio, Diego R, & Oliveira Jr, Osvaldo N. 2017. On the role of words in the network structure of texts: application to authorship attribution. *arXiv preprint arXiv:1705.04187*.

Bagnall, Douglas. 2015. Author identification using multi-headed recurrent neural networks. *arXiv preprint arXiv:1506.04891*.

Bagnall, Douglas. 2016. Authorship clustering using multi-headed recurrent neural networks. *arXiv preprint arXiv:1608.04485*.

Baraldi, Lorenzo, Grana, Costantino, & Cucchiara, Rita. 2015. A deep siamese network for scene detection in broadcast videos. *Pages 1199–1202 of: Proceedings of the 23rd ACM international conference on Multimedia*. ACM.

Bromley, Jane, Bentz, James W., Bottou, Léon, Guyon, Isabelle, LeCun, Yann, Moore, Cliff, Säckinger, Eduard, & Shah, Roopak. 1993. Signature Verification Using A "Siamese" Time Delay Neural Network. *IJPRAI*, **7**(4), 669–688.

Chopra, Sumit, Hadsell, Raia, & LeCun, Yann. 2005. Learning a similarity metric discriminatively, with application to face verification. *Pages 539–546 of: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE.

Gatt, Albert, & Krahmer, Emiel. 2017. Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. *arXiv preprint arXiv:1703.09902*.

Halvani, Oren, Winter, Christian, & Pflug, Anika. 2016. Authorship verification for different languages, genres and topics. *Digital Investigation*, **16**, S33–S43.

Halvani, Oren, Winter, Christian, & Graner, Lukas. 2017. Authorship Verification based on Compression-Models. *arXiv preprint arXiv:1706.00516*.

Jhamtani, Harsh, Gangal, Varun, Hovy, Eduard, & Nyberg, Eric. 2017. Shakespearizing Modern Language Using Copy-Enriched Sequence-to-Sequence Models. *arXiv preprint arXiv:1707.01161*.

Kabbara, Jad, & Cheung, Jackie Chi Kit. 2016. Stylistic Transfer in Natural Language Generation Systems Using Recurrent Neural Networks. *EMNLP 2016*, 43.

Kestemont, Mike. 2014. Function words in authorship attribution from black magic to theory? *Pages 59–66 of: Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLfL)@ EACL*.

Klimt, Bryan, & Yang, Yiming. 2004. The enron corpus: A new dataset for email classification research. *Machine learning: ECML 2004*, 217–226.

Koppel, Moshe, & Schler, Jonathan. 2004. Authorship verification as a one-class classification problem. *Page 62 of: Proceedings of the twenty-first international conference on Machine learning*. ACM.

Koppel, Moshe, & Winter, Yaron. 2014. Determining if two documents are written by the same author. *Journal of the Association for Information Science and Technology*, **65**(1), 178–187.

Koppel, Moshe, Schler, Jonathan, & Argamon, Shlomo. 2009. Computational methods in authorship attribution. *Journal of the Association for Information Science and Technology*, **60**(1), 9–26.

Koppel, Moshe, Schler, Jonathan, & Argamon, Shlomo. 2012a. Authorship Attribution: What's Easy and What's Hard. *JL & Pol'y*, **21**, 317.

Koppel, Moshe, Schler, Jonathan, Argamon, Shlomo, & Winter, Yaron. 2012b. The fundamental problem of authorship attribution. *English Studies*, **93**(3), 284–291.

Luyckx, Kim. 2011. *Scalability issues in authorship attribution*. ASP/VUBPRESS/UPA.

Luyckx, Kim, & Daelemans, Walter. 2008. Authorship attribution and verification with many authors and limited data. *Pages 513–520 of: Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics.

Myers, Isabel Briggs. 1962. The Myers-Briggs Type Indicator: Manual (1962).

Neculoiu, Paul, Versteegh, Maarten, Rotaru, Mihai, & Amsterdam, Textkernel BV. 2016. Learning Text Similarity with Siamese Recurrent Networks. *ACL 2016*, 148.

Queneau, Raymond. 1981. *Exercises in style*. Vol. 513. New Directions Publishing.

Somers, Harold, & Tweedie, Fiona. 2003. Authorship attribution and pastiche. *Computers and the Humanities*, **37**(4), 407–429.

Stamatatos, Efstathios. 2009. A survey of modern authorship attribution methods. *Journal of the American Society for information Science and Technology*, **60**(3), 538–556.

Stolerman, Ariel. 2015. *Authorship Verification*. Ph.D. thesis. Copyright - Copyright ProQuest, UMI Dissertations Publishing 2015; Last updated - 2015-05-12; First page - n/a.

Stolerman, Ariel, Overdorf, Rebekah, Afroz, Sadia, & Greenstadt, Rachel. 2011. Classify, but verify: Breaking the closed-world assumption in stylometric authorship attribution. *Page 23 of: IFIP Working Group*, vol. 11.

Verhoeven, Ben, Daelemans, Walter, & Plank, Barbara. 2016. TwiSty: A Multilingual Twitter Stylometry Corpus for Gender and Personality Profiling. *In: LREC*.

Zhu, Jianqing, Zeng, Huanqiang, Liao, Shengcai, Lei, Zhen, Cai, Canhui, & Zheng, LiXin. 2017. Deep Hybrid Similarity Learning for Person Re-identification. *arXiv preprint arXiv:1702.04858*.