Robust Authorship Verification with Transfer Learning

No Author Given

No Institute Given

Abstract. We address the problem of open-set authorship verification, a classification task that consists of attributing texts of unknown authorship to a given author when the unknown documents in the test set are excluded from the training set. We present an end-to-end model-building process that is universally applicable to a wide variety of corpora with little to no modification or fine-tuning. It relies on transfer learning of a deep language model and uses a generative adversarial network and a number of text augmentation techniques to improve the model's generalization ability. The language model encodes documents of known and unknown authorship into a domain-invariant space, aligning document pairs as input to the classifier, while keeping them separate. The resulting embeddings are used to train to an ensemble of recurrent and quasi-recurrent neural networks. The entire pipeline is bidirectional; forward and backward pass results are averaged. We perform experiments on four traditional authorship verification datasets, a collection of machine learning papers mined from the web, and a large Amazon-Reviews dataset. Experimental results surpass baseline and current state-of-the-art techniques, validating the proposed approach.

Keywords: authorship verification · transfer learning · language modeling.

1 Introduction

We investigate the applicability of transfer learning techniques to Authorship Verification (AV) problems, and propose a a method that uses some of the most recent advances in deep learning to achieve state of the art results on a variety of datasets. AV seeks to determine whether two or more text documents have been written by the same author. Some applications of AV include plagiarism analysis, sock-puppet detection, blackmailing, and email spoofing prevention [8]. Traditionally, studies on AV consider a closed and limited set of authors, and a closed set of documents written by such authors. During the training step, some of these documents (sometimes as long as a novel) are used. The goal can be formulated as to successfully identify whether the authors of a pair of documents are identical [15, 20, 12]. This type of AV tasks assumes access to the writing samples of all possible authors during the training step, which is not realistic. Recently, the AV problem has changed to reflect realistic –and more challenging– scenarios. The goal is no longer to individually learn the writing style of the authors (like in traditional AV methods), but to learn what differentiates two different authors within a corpus. This task involves predicting authorship of documents that may not have been previously encountered within the training set; in fact, the presence of the authors in the training data is not guaranteed either. That is, the test set may contain out of training

sample data; given a set of authors of unknown papers contained within the training data, $A_{train}^{unknown}$, and a set of authors of unknown papers in the test data, $A_{test}^{unknown}$, it is neither unreasonable nor unexpected to find that $A_{train}^{unknown} \cap A_{test}^{unknown} = \emptyset$.

Some other challenges arise in modern AV tasks, making authorship verification of a given pair of documents hard to infer. One is the lack of training data, which can manifest itself in any one or more of the following: the training set may be small, samples of available writings may be limited, or the length of the given documents may be insufficient. Another is the test and train documents belonging to different genre and/or topics, both within their respective sets as well as between the train and the test set –implying they were likely drawn from different distributions. The challenge is to ensure robustness in a multitude of possible scenarios. Regardless of the AV problem specifics, generally we assume a training dataset made of sets of triples:

$$D = (((x_{i,k}^{known}, x_{j,k}^{unknown}, y_{i,j,k})_{i=1}^{N})_{j=1}^{M})_{k=1}^{P}, \tag{1}$$

with $x_i \epsilon X_{known}, x_j \epsilon X_{unknown}$ a realization from random variables X_{known} and $X_{unknown}$, and the label $y_{i,j} \epsilon Y$ is drawn from a random variable Y, producing a total of P sets of realizations, each potentially by a different author, thus forming up to P source *domains*, because it can be argued that a collection of literary works by one author forms a latent domain of it's own. The goal is to learn a prediction function $f : X \rightarrow Y$ that can generalize well and make accurate predictions regarding documents written by authors both inside and outside of the training set, even if those documents were not seen in training. Less formally, in AV the task is composed of multiple sub-problems: for each given sub-set of texts, we are provided one or more documents that need to be verified and one or more that are known to be of identical authorship. We approach the AV problem by designing a straightforward deep document classification model that relies on transfer learning a deep language model, ensembles, an adversary, differential learning rates, and data augmentation. In order to ensure the design's versatility and robustness, we perform authorship verification on a collection of datasets that have little in common in terms of size, distribution, origins, and manner they were designed. For evaluation, we consider standard AV corpora with minimal amount of training data, PAN-2013 [13], PAN-2014E and PAN-2014N [28], PAN-2015 [29], a collection of scientific papers mined from the web [3], and Amazon Reviews dataset [9]. The proposed approach performs well in all scenarios with no specific modifications and minimal fine-tuning, defeating all baselines, PAN competition winners, as well as the recent Transformation Encoder and PRNN models that were recently shown to perform well on AV tasks. [9].

2 Method

Our method consists of three major components: augmentation, transfer learning, and the training/testing process itself. At a high level, we augment the data, fine-tune a deep LSTM-based language model (LM) known as ULMFit [10] on the augmented training set, train an ensemble of RNN and QRNN classifiers with the encoding produced by the LM forward and backward, and evaluate the test data while performing test-time data augmentation.

Table 1. Data augmentation techniques.

Augmentation Technique	Description	Test-time
Paragraph shuffle	Shuffle paragraphs in a document	Yes
Document splitting	Split document in others of varying sizes	Yes
Noise injection	Add noise using language model and adversary	Yes
Document matching	Find identical documents in other problems	Yes
Bidirectional models	Read texts forward and backward word by word	Yes
Document generation	Generate unknown document when authors don't match	No

2.1 Data Augmentation

We utilize various data augmentation techniques in order to improve model generalization (Table 1). They broadly fall into two categories, document manipulation and adversarial noise injection with the LM. In addition, most of these techniques can be applied to the test set documents during evaluation; however, some do more harm than good when used in such manner.

Noise injection is performed by a 5-layer LSTM model that was pre-trained on Wikipedia and fine-tuned on our data. In our setup, it acts as a generator with a 3-layer RNN classifier working as a critic. Adversarial loss function is a *weighted* average of the two losses

$$L_{GAN} = w_{avg}(L_{generator}(g) + L_{critic}(f \circ h)), \tag{2}$$

where g is the LM, f is an RNN and h is the linear classifier trained on RNN's average then max pooled and flattened 2 top layers. We use a weighted average because the nature of loss functions is very different. To improve quality of augmentation, we devised the following approach (Algorithm 1). Given a training set consisting of a number of problems, with each problem containing one or more documents known to be written by the same author and a single document of unknown authorship, we cycle through each problem in the training set. If for a given problem the ground truth answer is positive, we train on all documents and try injecting noise. If the critic can tell the fake, it means our new document is most likely too different from actual ones by this author to be of any use; we then try training some more, and inject shorter sentences and less of them. The process continues until critic is fooled or generator diverges – an unlikely event because critic is not hard to fool.

2.2 Transfer Learning

We hypothesize that documents form latent domains of their own based on various linguistic characteristics, making it beneficial to transform the pairs of documents into a domain-invariant space. Documents forming latent domains means that authorship verification is a separate but similar task for each domain. We cannot exploit the similarity between tasks directly because the data distributions are different, and not accounting for that while building a model would violate basic principles of machine learning [22]. Domain Adaptation (DA), a subset of Transfer learning, addresses such

```
Load pre-trained language model;
while Next problem do
    Load encoder layer;
    Unfreeze encoder layer for training;
    while Critic rejects document do
        Lower learning rate;
        Decrease generated sequence length;
        Decrease number of sequences generated;
        if ground_truth = YES then
            Train on all documents for one cycle;
            Inject a few short sentences into each document;
        else
            Train on known documents only;
            Inject a few short sentences into each known document;
            For the unknown document, generate a second one of equal length.
        end
    end
end
```

Algorithm 1: Noise injection algorithm using language model with an adversary

problems by establishing knowledge transfer from a labeled source domain to an unlabeled (or partially labeled) target domain, by exploring domain-invariant features or *invariants* which transfer across domains [23, 22, 6, 30], or by embedding the data into domain-invariant subspace. Another issue that we must address comes from the nature of the data. As the documents come in pairs, they are not readily suitable for standard classifiers. A naive approach of concatenation produces poor results, and various distance function schema suitable for most linear models are not very suitable for RNNs. To address these problems we utilize a deep language model that produces an encoder capable of producing an embedding representing a pair of documents. It also alleviates the need for data by being pre-trained on a large set of Wikipedia articles [10]. The domain discrepancy issue is in part mitigated too, because the resulting embedding subspace features are more invariant.

2.3 Network Architecture

In a gist, our model (Figure 1) is a bi-directional pipeline of recurrent neural networks. It is built on top of a pre-trained 5-layer LSTM model and takes it's last 3 (2 intermediate hidden ones and the final embedding output) layers as inputs by pooling them together. We use an ensemble of sequence classifiers, one based on an RNN and the other using a QRNN [2], a recent addition to the RNN family that combines some properties of recurrent and convolutional networks. Both are 3-layer models with the last 2 layers average then max pooled and passed through a ReLU non-linearity and then to logit units. We output probabilities rather than labels. The predictions made by RNN and QRNN are averaged.

In taking advantage of improved generalization through making the model bi-directional, we faced two challenges. First, the pre-trained LSTM model we used is uni-directional.

Second, QRNN design used in this paper does not support bi-directional training, either. We circumvented the issue by tokenizing and numericalizing the text data and first training in regular fashion on a normal pre-trained Wikipedia model, then loading the numericalized tokens backwards and using a model that was trained on Wikipedia backwards, as well. At test time, we reversed each document and gave the normal ones to the forward model and backward ones the backward version, then averaged the results of two runs, effectively reaping the benefits of using a bi-directional RNN without actually doing so.

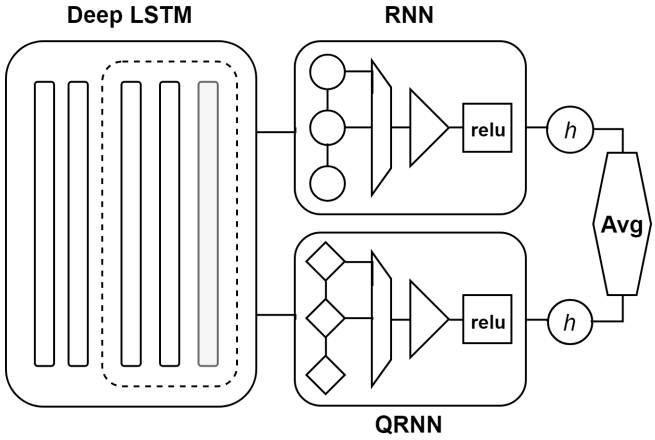

Fig. 1. Network Architecture: Deep LSTM outputs 2 hidden layer and final embedding, which are used as inputs by RNN and QRNN Ensemble. Both have 3 layers, top 2 are average then max pooled and passed through a ReLU layer. Each net uses a logit to make a probability prediction and results are averaged.

We call our design *2WD-UAV* in reference to ensembling of two versions of RNN for authorship verification and because of it's ULMFit heritage. The architecture is implemented in PyTorch with elements of fast.ai library [11].

3 Experiments

PAN We use all available authorship identification datasets released by PAN [1] (Table 2). Each PAN dataset consists of a training and test corpus and each corpus has a various number of distinct problems. Each problem is composed of one to five writings by a single person (implicitly disjoint For PAN2014 and PAN2015 and explicitly disjoint for PAN2013), and one piece of writing of unknown authorship. In the other words, we are given up to five pairs of documents where one document's authorship is known and the other one's is not. Two documents of a pair might be from significantly different genres

[1] http://pan.webis.de/data.html

Table 2. Dataset information

Dataset	Train	Test
PAN2013	10	30
PAN2014E	200	200
PAN2014N	100	200
PAN2015	100	500
Dataset	**Positive**	**Negative**
Amazon	4500	4500
MPLA*	720	720

Table 3. Similarity functions. x, y: document feature vectors, n: # of features in x and y

Metric	Description
Chi2 kernel	$\exp(-\gamma \sum_i [\frac{(x_i - y_i)^2}{(x_i + y_i)}])$
Cosine similarity	$xy^T / (\|\|x\|\|\|\|y\|\|)$
Euclidean	$\sqrt{\sum_i (x_i - y_i)^2}$
Linear kernel	$x^T y$
RBF kernel	$\exp(-\gamma \|\|x - y\|\|^2)$
Mean of L1 norm	$\sum_i^n \|x_i - y_i\|/n$
Sigmoid kernel	$\tanh(\gamma x^T y + c_0)$

and topics. The length of a document changes from a few hundred to a few thousand words. PAN2014 includes two datasets: Essays and Novels. The paired documents in PAN datasets are used for our experiments. For a problem $P = (S, T)$, S (source) is the first document and T (target) is the second document of a PAN problem [9].

Amazon Reviews We use a dataset made by selecting 300 authors with at least 40 reviews to make the positive and negative candidate sets. Then, for each author, the positive candidate set is all possible and unique combinations of the author's reviews. A positive class consists of 4500 review pairs from this positive candidate set at random. The negative candidate set is made of all unique and possible combinations of review pairs having different authors. For this dataset, the negative class of equal size with the positive class was created by random selection from the negative candidate set. In prior work, 5-fold cross validation was used for this data. We do the same in order for our results to be comparable. [9].

MLPA* This schema was created using MPLA-400 dataset that contains 20 articles by each of the top-20 authors by citation in Machine Learning [3]. In MLPA*, only publications from MPLA-400 that are written by a single author and have no co-authors are used [9]. To keep the distribution of authors and classes balanced, MPLA* contains an equal number of single-authorship articles from all existing 20 authors. The positive class consists of the pairs which are made up of all possible combinations of same-authorship articles ($20 \times \binom{9}{2} = 720$). The negative class includes the pairs that are randomly selected from the set of all unique combinations of articles of different authorship and is of the same size as the positive class. Like Amazon Reviews, MLPA* dataset authors recommend using 5-fold cross validation [9].

3.1 Baselines

We compare our method with the top methods of PAN AV competition between 2013 and 2015 (Table 2). The results of each method for one year of the competition are available and we report them here. Our comparisons are not impacted by different parameter settings and implementation details of these methods as long as we keep the test and training sets the same as theirs.

We choose several classifiers widely used in the area with the seven similarity measures to set strong baselines (Table 3). Since each example in our underlying dataset structure

comprises two documents, we need to adapt it to the structure of an ordinary classifier input by converting them to one single entity. A simple direct way is to concatenate their feature vectors. However, our experiments show it provides weak results mostly equal to the random label assignment. So, we define the *summary vector* as a single unit representative of each example/problem $P = (D^S, D^T)$ by utilizing several similarity measures. The summary vector comprises a class of several metrics each measuring one aspect of the closeness of the two documents (D^S and D^T) of the pair for all underlying feature sets. For any two feature vector documents x, y their summary vector is $sum(x, y) = [sim_i^j(x, y)]$ where $sim_i^j(x, y)_{1 \leq i \leq M, 1 \leq j \leq F}$ computes the ith similarity metric of M metrics in Table 3 under jth of $F = 7$ feature sets (Section 3.2) between x, y. Then, we use a suite of classifiers including **SVM**, Gaussian Naive Bayes (**GNB**), K-Nearest Neighbor (**KNN**), Logistic Regression (**LR**), Decision Tree (**DT**) and Multi-Layer Perception (**MLP**) to predict the class label. All baselines are implemented by the scikit-learn library [24].

3.2 Experimental Settings

2WD-UAV For our model, a number of important parameters are set. Most importantly, to achieve our results, we make use of recent work on alternating learning rates, as well as one-cycle learning policy [26, 27]. The basic approach to training is as follows:

- Contract learning rate lr for one cycle
- Freeze it and save
- Give the learning rate on next layer a very large value
- Freeze it and save unfreeze the previous one
- Assign a very small value to the next layer
- Continue cycling until gradients explode
- Return the last saved checkpoint – this is the global minimum

We also use a range of momentum across layers, as well as different learning rates for each. For the optimizer we choose AdamW [19], an improved version of Adam [14] with better weight decay regularization. We begin with weight decay of 0.03 and regularize by adjusting it as training progresses.

Baselines All documents of D^S and D^T are represented in vector space model under several feature sets with term frequency and Boolean feature value assignment separately. Seven feature sets are used: unigram, bigram, 3-gram, 4-gram, unigram Part Of Speech (POS), bigram POS, and char-5gram [2]. Gaussian distribution is chosen for Naive Bayes. For K-Nearest Neighbor we set K=3. The L-2 regularization is used for Logistic Regression. For document expansion, we set the size of the sliding window to $l = 10$. On average it expands one document into 30 smaller documents for PAN datasets. All other parameters are selected based on pilot experiments. We report accuracy, the Area Under Receiver Operating Characteristic (ROC) curve [4] (AUC). The higher AUC and Score indicate more effective classification.

[2] we use scikit-learn software for all linguistic features

4 Results and Discussion

Table 4. Results on PAN datasets.

		PAN14E			PAN14N		
Category	Method	Acc.	ROC	Score	Acc.	ROC	Score
Baseline	GNB	0.675	0.741	0.5	0.56	0.743	0.416
Baseline	LR	0.675	0.728	0.491	0.515	0.604	0.311
Baseline	MLP	0.7	**0.768**	0.538	0.54	0.782	0.422
PAN	FCMC	0.58	0.602	0.349	0.71	0.711	0.508
PAN	Frery	0.71	0.723	0.513	0.59	0.61	0.36
	TE	0.67	0.675	0.452	0.695	0.7	0.487
Our method	**2WD-UAV**	**0.73**	0.761	**0.555**	0.68	**0.801**	**0.552**
		PAN13			PAN15		
Category	Method	Acc.	ROC	Score	Acc.	ROC	Score
Baseline	GNB	0.633	0.795	0.503	0.552	0.78	0.431
Baseline	LR	0.7	0.781	0.547	0.544	0.796	0.433
Baseline	MLP	0.533	0.5	0.267	0.554	0.687	0.381
PAN	MRNN	-	-	-	**0.76**	0.81	0.61
PAN	Castro	-	-	-	0.69	0.75	0.52
PAN	GenIM	**0.8**	0.792	**0.633**	-	-	-
PAN	CNG	-	**0.842**	-	-	-	-
	TE	0.8	0.835	0.668	0.748	0.75	0.561
Our method	**2WD-UAV**	**0.82**	0.825	**0.677**	0.75	**0.822**	**0.617**

We compare our proposed model 2WD-UAV with several relevant baselines. Table 4 evaluates our model with PAN datasets for different years and also the best performing model in the relevant competition years for PAN. Results show that 2WD-UAV consistently outperforms all baselines and all best-reported models in PAN competitions for all years in the Score metric. The Score metric is essentially $Accuracy \times ROC$ thereby measuring joint performance gains as both ROC and accuracy are important. 2WD-UAV outperforms in Accuracy for all competitors in PAN14Essay and PAN13 dataset. It is the second best in PAN15 just offset by one decimal point. While it is not performing the best in accuracy for PAN14Novels, it yields competitive performances of accuracy and outperforms all others in ROC metric. 2WD-UAV also outperforms all other models in the ROC metric for PAN15. For PAN14E and PAN 13, it outperforms several baselines and offers stellar performance in ROC metric, just to be second to MLP and CNG respectively. While it is true that the proposed approach is not always the best performing on PAN data in every metric except Score, we believe one reason is due to the inherently smaller data sizes (both total words of data per author to train upon and also the total number of authors to scale up training) that make the approach a little weak. Hence, we further explored larger datasets of Amazon Reviews [9] and MLPA* [9, 3] in Table 5 which shows significant performance gains in accuracy, defeating a variety of baselines. All in all, we do find stable and consistent performance gains with

Table 5. Accuracy using 5-fold cross-validation on MLPA* and Amazon Reviews.

Dataset	Methods							
	2WD-UAV	PRNN	SVM	NB	LR	KNN	DT	MLP
MLPA*	**0.766**	0.703	0.621	0.635	0.671	0.64	0.28	0.686
Amazon Reviews	**0.941**	0.922	0.818	0.741	0.839	0.831	0.818	0.858

2WD-UAV across a variety of datasets and baselines and best performing competitors in PAN. All these indicate that the proposed approach as a robust and consistent in improving authorship verification beyond the existing state-of-the-art.

5 Related Work

Domain Adaptation Documents forming latent domains means that authorship verification is a separate but similar task for each domain. We cannot exploit the similarity between tasks directly because the data distributions are different, and not accounting for that while building a model would violate basic principles of machine learning [22]. Domain Adaptation (DA) addresses such problems by establishing knowledge transfer from a labeled source domain to an unlabeled (or partially labeled) target domain, and by exploring domain-invariant features or *invariants* which transfer across domains [23, 22, 6, 30].

Authorship Verification In vast majority of the AV approaches, the writing style of a questioned author is known to us as we are given some scripts of the author and the task is to determine whether a piece of work is written by the same person. The depth of difference between two sets of documents is measured using the unmasking technique while ignoring the negative examples [15]. This one-class technique achieves high accuracy for 21 considerably large books (ebook above 500K). A simple feed forward three-layer neural network, an auto-encoder, is used for AV considering it a one-class classification problem [20]. They observe the behavior of the neural network for documents by different authors and build a classifier for each author. Their idea originates from one of the first applications of auto-encoder in classification as a novelty detector [12]. AV is also studied for detecting sock-puppets who deliberately change their writing styles to pass the filters and provide opinion Spam. A spy induction method is proposed to leverage the test data in training step under "out-of-training" setting [8] where a questioned author is from a closed set of candidates while appearing unknown to the verifier. However, in a more realistic case we have no specified writing samples of a questioned author and there is no closed candidate set of authors. Since 2013, a surge of interest arose for this type of AV problem. [25] investigate whether a document is one of the outliers in a corpus by generalizing the Many-Candidate method by [16]. The best method of PAN 2014 for Essays dataset optimizes a decision tree. Its method is enriched by adopting variety of features and similarity measures [5]. However, for the Novels dataset, the other dataset of that year, the best results are achieved by an author verifier using fuzzy C-Means clustering [21]. In an alternative approach, [17] generate a set of impostor documents and apply iterative feature randomization to compute the

similarity distance between pairs of documents. One of the more interesting and powerful approaches investigates the language model of all authors using a shared recurrent layer and builds a classifier for each author [1]. Parallel recurrent neural network and transformation auto-encoder approaches were recently shown to produce excellent results far a variety of AV problems [9]. The AV problem is also studied by a non Machine Learning model comprising of a compression algorithm, a dissimilarity method and a threshold. When evaluated on PAN datasets, this approach stands at first ranking position for the two out of four PAN datasets [7]. Recently, Linguistic traits of sock-puppets are deeply studied to verify the authorship of a pair of accounts in online discussion communities [18].

6 Conclusion

Authorship verification has always been a challenging problem. It can be even more difficult when no writing samples of questioned author/authors is given. In this paper, we explore the possibility of a more general approach to the problem, one that does not rely on having most of the authors within the training set. To this end, we use transfer and adversarial learning learning, data augmentation, ensemble methods, and cutting edge developments in training deep models to produce an architecture that is to the best of our knowledge novel at least to problem setting. Our design exhibits a high degree of robustness and stability when dealing with out-of-sample (previously unseen) authors and lack of training data and delivers state-of-the-art performance.

References

1. Bagnall, Douglas: Author identification using multi-headed recurrent neural networks. arXiv preprint arXiv:1506.04891 **60**(1), 9–26 (2009)
2. Bradbury, J., Merity, S., Xiong, C., Socher, R.: Quasi-Recurrent Neural Networks. International Conference on Learning Representations (ICLR 2017) (2017)
3. Dainis Boumber, Y.Z., Mukherjee, A.: Experiments with convolutional neural networks for multi-label authorship attribution. In: chair), N.C.C., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S., Tokunaga, T. (eds.) Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Paris, France (2018)
4. Egan, J.P.: Signal detection theory and {ROC} analysis. Academic Press (1975)
5. Frery, J., Largeron, C., Juganaru-Mathieu, M.: Ujm at clef in author verification based on optimized classification trees. In: Conference and Labs Evaluation Forum 2014. p. 7p (2014)
6. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. The Journal of Machine Learning Research **17**(1), 2096–2030 (2016)
7. Halvani, O., Winter, C., Graner, L.: On the usefulness of compression models for authorship verification. In: Proceedings of the 12th International Conference on Availability, Reliability and Security. p. 54. ACM (2017)
8. Hosseinia, M., Mukherjee, A.: Detecting sockpuppets in deceptive opinion spam. Computing Research Repository **abs/1703.03149** (2017)

9. Hosseinia, M., Mukherjee, A.: Experiments with neural networks for small and large scale authorship verification (2018)
10. Howard, J., Ruder, S.: Fine-tuned language models for text classification. CoRR **abs/1801.06146** (2018), http://arxiv.org/abs/1801.06146
11. Howard, J., et al.: fastai. https://github.com/fastai/fastai (2018)
12. Japkowicz, N., Myers, C., Gluck, M., et al.: A novelty detection approach to classification. In: IJCAI. vol. 1, pp. 518–523 (1995)
13. Juola, P., Stamatatos, E.: Overview of the author identification task at pan 2013. In: CLEF (Working Notes) (2013)
14. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. ArXiv (2014)
15. Koppel, M., Schler, J.: Authorship verification as a one-class classification problem. In: Proceedings of the Twenty-First International Conference on Machine learning. p. 62. Association for Computing Machinery (2004)
16. Koppel, M., Schler, J., Argamon, S.: Authorship attribution in the wild. Language Resources and Evaluation **45**(1), 83–94 (2011)
17. Koppel, M., Winter, Y.: Determining if two documents are written by the same author. In: Journal of the Association for Information Science and Technology. vol. 65, pp. 178–187. Wiley Online Library (2014)
18. Kumar, S., Cheng, J., Leskovec, J., Subrahmanian, V.: An army of me: Sockpuppets in online discussion communities. In: Proceedings of the 26th International Conference on World Wide Web. pp. 857–866. International World Wide Web Conferences Steering Committee (2017)
19. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in adam. arXiv preprint arXiv:1711.05101 (2017)
20. Manevitz, L., Yousef, M.: One-class document classification via neural networks. In: Neurocomputing. vol. 70, pp. 1466–1481. Elsevier (2007)
21. Modaresi, P., Gross, P.: A language independent author verifier using fuzzy c-means clustering. In: Conference and Labs Evaluation Forum (Working Notes). pp. 1084–1091 (2014)
22. Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: International Conference on Machine Learning. pp. 10–18 (2013)
23. Pan, S.J., Yang, Q.: A survey on transfer learning. Knowledge and Data Engineering, IEEE Transactions on **22**(10), 1345–1359 (2010)
24. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al.: Scikit-learn: Machine learning in python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
25. Seidman, S.: Authorship verification using the impostors method. In: Conference and Labs Evaluation Forum 2013 Evaluation Labs and Workshop–Working Notes Papers. pp. 23–26. Citeseer (2013)
26. Smith, L.N.: A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. Computing Research Repository **abs/1803.09820** (2018), http://arxiv.org/abs/1803.09820
27. Smith, L.N., Topin, N.: Super-convergence: Very fast training of residual networks using large learning rates. CoRR **abs/1708.07120** (2017)
28. Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P., Sanchez-Perez, M.A., Barrón-Cedeño, A.: Overview of the author identification task at pan 2014. In: CLEF 2014 Evaluation Labs and Workshop Working Notes Papers, Sheffield, UK, 2014. pp. 1–21 (2014)
29. Stamatatos, E., Potthast, M., Rangel, F., Rosso, P., Stein, B.: Overview of the pan/clef 2015 evaluation lab. In: International Conference of the Cross-Language Evaluation Forum for European Languages. pp. 518–538. Springer (2015)

30. Zhao, H., Zhang, S., Wu, G., Costeira, J.P., Moura, J.M.F., Gordon, G.J.: Multiple source domain adaptation with adversarial training of neural networks. Computing Research Repository **abs/1705.09684** (2017), http://arxiv.org/abs/1705.09684