Cross-Domain Authorship Attribution Based on Compression
Notebook for PAN at CLEF 2018

Oren Halvani* and Lukas Graner

Fraunhofer Institute for Secure Information Technology SIT
Rheinstrasse 75, 64295 Darmstadt, Germany
{FirstName.LastName}@SIT.Fraunhofer.de

Abstract Authorship attribution (AA) is a very well studied research subject and the most prominent subtask of authorship analysis. The goal of AA is to identify the most likely author of an anonymous document among a set of known candidate authors, for which sample documents exist. Even after more than a century of intensive research, AA is still far from being solved. One open question, for example is, if the goal of AA can be successfully achieved, if the anonymous document and the known sample documents come from different domains such as genre or topic. We present a lightweight authorship attribution approach named COBAA ("Compression-Based Authorship Attribution") which is an attempt to answer this question. COBAA is based solely on a compression algorithm as well as a simple similarity measure and does not involve a training procedure. Therefore, the method can be used out-of-the-box even in real-world scenarios, where no training data is available. COBAA has been evaluated at the PAN 2018 Author Identification shared task and was ranked third among 11 participating approaches. The method achieved 0.629 in terms of Mean Macro-F1 on a corpus with attribution problems, distributed across five languages (English, French, Italian, Polish and Spanish).

1 Introduction

Attributing an anonymous text to its most likely author is a very well-studied problem, which dates back to the 19th century [19]. Even after more than ten decades, the problem is still far from being solved and has become an important research subject, across many fields and domains. The discipline that concerns itself with this problem is known as **authorship attribution**[1] (AA), which is a subdiscipline of authorship analysis.

There are two types of AA problems: *closed-set* and *open-set*, where the former assumes that the candidate set is closed and thus contains sample writings of the true author of the unknown document. Here, the task is to compare the unknown document

* Corresponding author.

[1] Over the past, a number of synonyms for AA appeared in the literature including: authorship recognition [1], authorship determination [6], authorship classification [7], person identification [8], authorship de-identification [12] or author identification [21].

to each of the writings in the candidate set and to output the author behind the document, which is stylistically most similar to the unknown document. The majority of existing research focuses on this case [28]. In contrast, the *open-set* case considers a more realistic setting, where the true author is not longer believed to be present in the candidate set. In case of uncertainty, an *open-set* AA method can then output a "*don't know*" response, instead of a wrong author of a text that is stylistically most similar to the unknown document. Koppel et al., for example, follow this approach [16].

So far, many different types of machine learning models have been applied to solve AA, including SVMs [8], neural networks [3,12,14,15], LDA [30]. The common denominator of these is that they rely on explicitly defined features (or more precisely, feature vectors) that serve as an input for the chosen machine learning model (see Figure 1). The most commonly used features in AA are character n-grams, frequent tokens (such as function words) and POS tags.

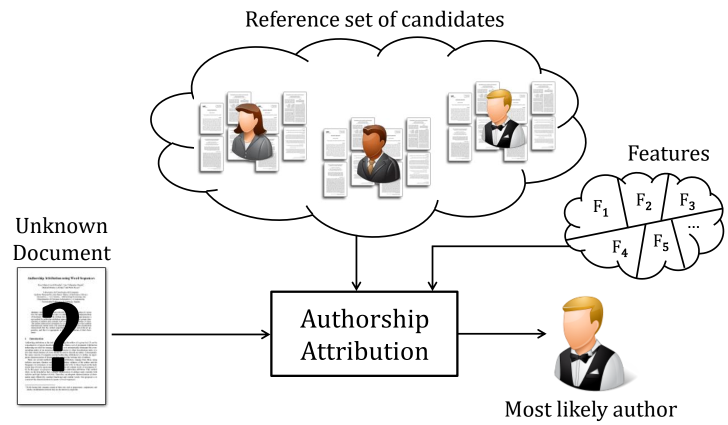

Figure 1. A simplified (closed-set) authorship attribution scheme.

An alternative approach to these are AA methods that are based on compression models. The biggest advantage of these approaches is that instead to define features explicitly, the entire feature extraction process is delegated to an underlying compression algorithm. In the context of AA, compression algorithms have been explored in a number of previous research works including [2,17,18,22,23,26]. According to the reviewed literature, the most frequently employed algorithm in compression-based AA approaches is PPM[2], which we also use in our approach. PPM is an adaptive statistical data compression technique proposed by Cleary and Witten [4] and has shown promising results, not only in existing authorship attribution but also in authorship verification approaches ([10,11,29]) as well as other text classification tasks.

[2] PPM stands for "***P**rediction by **P**artial **M**atching*".

PPM makes use of a statistical model, by computing symbol probabilities and respectively encoding symbols one by one. This model records number of occurrences and probabilities for each symbol $\sigma \in \mathbb{S}$, following a specific context C i. e., a preceding sequence of symbols. The context length is variable, although many PPM implementations are limited by an upper bound, which is referred to as "order" $\mathcal{O}$. As a consequence, only context lengths ranging from 0 (meaning the zero length context ") to $\mathcal{O}$ are considered. Essentially, the PPM model is a set of tables $\mathcal{T}_\upsilon = \{\mathcal{T}_{\upsilon,C_1}, \mathcal{T}_{\upsilon,C_2}, \dots\}$, where υ denotes the context length and $\mathcal{T}_{\upsilon,C_i} = \{(\sigma, \#(\sigma, C), P(\sigma|C)) \mid C = (c_\upsilon, c_{\upsilon-1}, \dots c_1)\}$ a subtable, which comprises symbol probabilities for a specific context C_i. Here, $\#(\sigma, C)$ indicates the occurrences of σ follow after C and P the probability.

In the literature, many variants of the core PPM algorithm exist, where the most common are PPM**a** and PPM**b** [4], PPM**c** [20], PPM**d** and PPM* [5]. Apart from PPM*, all introduce an order, but can be distinguished in the way how the probabilities $P(\sigma|C)$ are calculated. In PPM**d**[3], which is the variant we use in our approach, the probability computation is performed by $P(\sigma|C) = (2 \cdot \#(\sigma, C) - 1)/(2\alpha)$, where α is the number of distinct symbols that are present in the subtable that corresponds to the context C.

It should be highlighted that all probabilities in each subtable have to sum up to 1. To ensure this, an additional escape symbol Esc is introduced that exists in each subtable $\mathcal{T}_{\upsilon,C}$ by default, where its probability is $P(Esc|C) = 1 - \sum_{\sigma\in\mathbb{S}} P(\sigma|C)$. Within each subtable $\mathcal{T}_{\upsilon,C}$ the escape symbol represents all other symbols that have not occurred after the context $C = (c_\upsilon, c_{\upsilon-1}, \dots, c_1)$. By this, Esc acts as a fallback entry that points to the subtable $\mathcal{T}_{\upsilon-1,C'}$ where C' is the shortened context $(c_{\upsilon-1}, c_{\upsilon-2}, \dots, c_1)$. In this way, the resulting linkage can be thought of a tree-like data structure, where each node represents a subtable. The probability for a σ can therefore be tracked down in the subtables corresponding to the shortened contexts. In the case that not even $\mathcal{T}_{0,('')}$ contains σ, we assume for each $\sigma \in \mathbb{S}$ an equal probability of $\frac{1}{|\mathbb{S}|}$.

At the compression process, each input symbol is compressed successively, where for each one two steps are made, encoding and table updating. In the first step, the symbol is encoded via arithmetic coding *AC*, more precisely via adaptive *AC*, since the probability distribution is constantly changing as it is dependent on the current PPM model and subtable. More precisely, the distribution is composed of the probabilities of all symbols in the subtable corresponding to the given context. If the symbol σ exists the encoding is completed by simply encoding this symbol for the aforementioned probability distribution. Otherwise, the escape symbol is encoded and the process is repeated for the subtable corresponding to the shortened context, until σ is found and encoded. The second step is updating the PPM model tables. The occurrences of σ are incremented in all subtables corresponding to the original context and all its shortened versions. This also changes the recorded probabilities as a consequence.

The following example illustrates the compression of the word *senses* given the PPMd implementation with $\mathcal{O} = 2$, after the substring ***sense*** has already been encoded. The

[3] To our best knowledge, PPMd is the most widely used variant of PPM, not only in various research domains but also in commercial and open-source compression implementations.

current statistical probability model and its tables are shown in Table 1. As a first step, the symbol 's' with the given context 'se' needs to be encoded. Since there is no entry yet for 's' in $\mathcal{T}_{2,(\mathrm{se})}$ and $\mathcal{T}_{1,(\mathrm{e})}$ but there is one in $\mathcal{T}_{0,('')}$ the escape symbol is encoded two times and afterwards symbol 's', each regarding the probability distributions given by the subtables, respectively. As an overview Figure 2 shows the aggregated result of the encoding of the three symbols Esc, Esc and 's'. The highlighted area represents the final *AC* encoded interval of the given symbol 's'. For the next step, the occurrences and probabilities in the subtables are updated, as highlighted in Table 1.

$\mathcal{T}_2$			
C	σ	**Count**	**Prob.**
en	s	1	$1/2$
	Esc		$1/2$
ns	e	1	$1/2$
	Esc		$1/2$
se	n	1	$1/2 \rightarrow 1/4$
	s	$\rightarrow 1$	$\rightarrow 1/4$
	Esc		$1/2 \rightarrow 1/2$

$\mathcal{T}_1$			
C	σ	**Count**	**Prob.**
e	n	1	$1/2 \rightarrow 1/4$
	s	$\rightarrow 1$	$\rightarrow 1/4$
	Esc		$1/2 \rightarrow 1/2$
n	s	1	$1/2$
	Esc		$1/2$
s	e	2	$3/4$
	Esc		$1/4$

$\mathcal{T}_0$			
C	σ	**Count**	**Prob.**
	e	2	$3/10 \rightarrow 3/12$
	n	1	$1/10 \rightarrow 1/12$
	s	$2 \rightarrow 3$	$3/10 \rightarrow 5/12$
	Esc		$3/10 \rightarrow 3/12$

Table 1. PPM tables for the word ***senses*** at the step of adding the last symbol ***s***

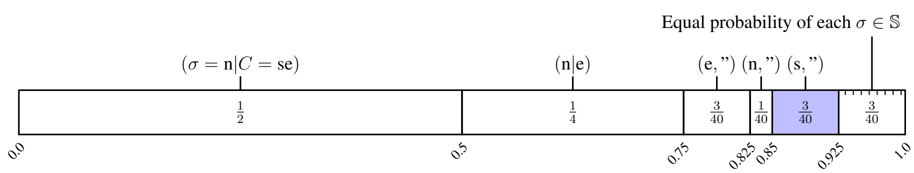

Figure 2. Aggregated probability distribution for the arithmetic coding to encode symbol ***s***

2 Proposed Approach

In the following, we present our lightweight AA scheme COBAA ("**Co**mpression-**B**ased **A**uthorship **A**ttribution"), which is almost entirely based on our already published authorship verification approach COAV [11]. First, we introduce a compact notation used along this section. Next, we mention which prerequisites are required to reproduce our approach, which is then explained in detail.

2.1 Notation

In the context of the PAN-2018 AA task [13], an attribution problem is defined as $p = (\mathbb{U}, \mathbb{D}_{candidates})$. Here, $\mathbb{U} = \{\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_\ell\}$ denotes a set of ℓ documents of unknown authors and $\mathbb{D}_{candidates} = \{\mathbb{D}_{\mathcal{A}_1}, \mathbb{D}_{\mathcal{A}_2}, \ldots, \mathbb{D}_{\mathcal{A}_n}\}$ a set of document collections of n known candidate authors $\mathbb{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n\}$. Each document collection of an author $\mathcal{A}_i$ is defined as $\mathbb{D}_{\mathcal{A}_i} = \{\mathcal{D}_{1A_i}, \mathcal{D}_{2A_i}, \ldots, \mathcal{D}_{mA_i}\}$. The PAN-2018 AA task focuses on a closed set attribution problem. Therefore, the task is to determine the true author $\mathcal{A}_x \in \mathbb{A}$ of each unknown document $\mathcal{U}_j \in \mathbb{U}$.

2.2 Prerequisites

As a first prerequisite, we use the PPMd compression algorithm. To avoid reinventing the wheel by reimplementing PPMd from scratch, we used the existing compression library *SharpCompress*[4]. As stated earlier in this paper, our approach does not require any type of training. However, this is only true, because we used the default parametrization regarding the PPM compressor, which is hard coded in the involved C# library. In fact, there are two tweakable parameters (`ModelOrder` and `AllocatorSize`). Based on the observations we explained in a previous PAN shared task [9], we decided to omit both hyperparameters by using the default parametrization (`ModelOrder` = 6 and `AllocatorSize` = 2^{24}).

As a second prerequisite, we require a measure that is able to determine the similarity between the resulting compressed documents. For this, we decided to use the CBC[5] measure, which has been proposed by Sculley and Brodley [27]. We refer the interested reader to our previous work [11] to gain a better understanding regarding this decision. The CBC measure is defined as:

$$\mathrm{CBC}(x, y) = 1 - \frac{C(x) + C(y) - C(xy)}{\sqrt{C(x)C(y)}}, \tag{1}$$

where x and y represent two documents and $C(\cdot)$ the length of a compressed document. It should be highlighted that the CBC function maps into the interval $[0; 1]$. However, it is not a metric as it violates the triangle inequality. Based on PPMd and CBC, our approach is explained in the following subsections.

2.3 Data representation

Inspired by the "profile-based AV method", proposed by Potha and Stamatatos [24], we decided also to follow the profile-based paradigm. Therefore, we first concatenate all sample documents in $\mathbb{D}_{\mathcal{A}_i}$ into a single document $\mathcal{D}_{\mathcal{A}_i} = \mathcal{D}_{1A_i} \circ \mathcal{D}_{2A_i} \circ \ldots \circ \mathcal{D}_{mA_i}$. As a result, the candidate author $\mathcal{A}_i$ is represented by only one known document $\mathcal{D}_{\mathcal{A}_i}$. This procedure is applied for all authors in $\mathbb{A}$ such that we end up with n known documents $\mathcal{D}_{\mathcal{A}_1}, \mathcal{D}_{\mathcal{A}_2}, \ldots, \mathcal{D}_{\mathcal{A}_n}$. Given PPMd, we compress each known document $\mathcal{D}_{\mathcal{A}_i}$ and each unknown document $\mathcal{U}_j$ into their compressed representation $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_\ell$, respectively.

[4] Offered by Adam Hathcock: https://github.com/adamhathcock/sharpcompress
[5] CBC stands for "***C****ompression-****B****ased* ***C****osine*".

2.4 Computing Similarities

Once all documents have been compressed, we compute similarities via CBC$(\cdot,\cdot)$ for all $n \cdot \ell$ document pairs $\{(X_i, Y_j)|(i \in \{1,2,\ldots,n\}) \wedge (j \in \{1,2,\ldots,\ell\})\}$. Each unknown document $\mathcal{U}_j$ (more precisely, its compressed representation Y_j) receives a list $\mathcal{S}_j \in (\mathbb{A} \times \{s_1, s_2, \ldots, s_n | s_i = \text{CBC}(X_i, Y_j)\})$, which consists of candidate authors and similarity scores regarding their corresponding known documents.

2.5 Decision

To determine the most likely author for each $\mathcal{U}_j \in \mathbb{U}$, we sort each corresponding list $\mathcal{S}_j$ regarding the similarities in descending order and pick out the first tuple $(s_q, \mathcal{A}_r) \in \mathcal{S}_j$. Here, $\mathcal{A}_r$ represents the author, whose document $\mathcal{D}_{A_r}$ is most similar to the unknown document $\mathcal{U}_j$, in terms of "writing style".

3 Evaluation

In the following subsections, we describe our evaluation, where we first explain which corpora and baselines were considered.

3.1 Corpora

Since COBAA does not involve trainable (hyper-)parameters, we were in the fortunate position to benefit from "two evaluation" corpora:

1. The provided **training** corpus $\mathcal{C}_{\textit{Train}}$:
"*pan18-cross-domain-authorship-attribution-training-dataset-2017-12-02*" [13].
2. The official (hidden) **evaluation** corpus $\mathcal{C}_{\textit{Eval}}$:
"*pan18-cross-domain-authorship-attribution-test-dataset2-2018-04-20*" [13].

$\mathcal{C}_{\textit{Train}}$ contains 10 problems $p_1, p_2, \ldots, p_{10}$, where each problem pair (p_{2i-1}, p_{2i}) belongs to the same language $\mathcal{L} \in \{\text{English, French, Italian, Polish, Spanish}\}$. Let $\mathbb{A}_{2i-1}$ and $\mathbb{A}_{2i}$ denote the set of candidate authors of p_{2i-1} and p_{2i}, respectively. Each $\mathbb{A}_{2i}$ is a quarter the size of $\mathbb{A}_{2i-1}$, which has as implication regarding the attribution results, presented in the next subsection.

3.2 Results on the Training Corpus

The first results we present are regarding the provided training corpus $\mathcal{C}_{\textit{Train}}$, which we used as an additional evaluation corpus. Besides COBAA, we also applied the provided SVM-baseline[6] on $\mathcal{C}_{\textit{Train}}$. The results for both are given in Table 2, where it can be seen that the baseline performs much better than a random guess (one hit out of n possible candidate authors). However, COBAA seems to be more effective, as (with the exception of p_7) it was able to outperform the SVM-baseline regarding any other

Table 2. Results regarding the **training** corpus $\mathcal{C}_{Train}$.

	Problem	**Language**	**Macro-F$_1$**	**Macro-Precision**	**Macro-Recall**	**Micro-Accuracy**
Baseline	p_1	English	0.426	0.428	0.537	0.552
	p_2	English	0.588	0.624	0.683	0.619
	p_3	French	0.607	0.646	0.684	0.633
	p_4	French	0.820	0.820	0.870	0.762
	p_5	Italian	0.508	0.511	0.623	0.662
	p_6	Italian	0.517	0.558	0.630	0.717
	p_7	Polish	0.437	0.455	0.515	0.485
	p_8	Polish	0.822	0.800	0.878	0.867
	p_9	Spanish	0.612	0.623	0.697	0.684
	p_{10}	Spanish	0.636	0.652	0.641	0.719
		average$(\cdot)$ $=$	**0.597**			
Our approach	p_1	English	0.523	0.545	0.659	0.638
	p_2	English	0.734	0.715	0.767	0.857
	p_3	French	0.635	0.708	0.685	0.673
	p_4	French	0.896	0.883	0.940	0.857
	p_5	Italian	0.582	0.580	0.744	0.588
	p_6	Italian	0.595	0.606	0.825	0.717
	p_7	Polish	0.420	0.507	0.478	0.427
	p_8	Polish	0.789	0.780	0.800	0.933
	p_9	Spanish	0.709	0.736	0.773	0.744
	p_{10}	Spanish	0.779	0.773	0.788	0.844
		average$(\cdot)$ $=$	**0.666**			

problem, in terms of Macro-F_1. A closer look on the **third** column in Table 2 reveals that the resulting Macro-F_1 score for each problem p_{2i} is higher than those of p_{2i-1}. This applies for both the SVM-baseline and COBAA. The most likely explanation for this is that the number of candidates in p_{2i-1} is smaller than those of p_{2i}. More precisely, each p_{2i-1} contains 20, while for p_{2i} there are 5 candidate authors. Another observation than can be made from Table 2 relates to the columns Problem, Language and Macro-F_1. In particular, one can see several significant differences regarding p_{2i-1} and p_{2i} and their corresponding languages. For example, regarding Polish, the differences are quite large (0.385 for the baseline and 0.369 for COBAA). Similarly, for French the differences are 0.213 (baseline) and 0.261 (COBAA). In contrast to both languages, for Italian the differences are minimal 0.009 (baseline) and 0,013 (COBAA). However, at the present time we do not have a reasonable explanation for this observation, which we therefore leave as a subject for future work.

3.3 Competition Results

The second results are based on the official PAN 2018 competition, where COBAA was evaluated among 11 submitted approaches. The results are given in Table 3. As can be

Rank	Participant	Mean Macro-F_1	Runtime
1	custodio18	0.685	00:04:27
2	murauer18	0.643	00:19:15
3	**halvani18**	**0.629**	**00:42:50**
4	mosavat18	0.613	00:03:34
5	yigal18	0.598	00:24:09
6	delcamporodriguez18	0.588	00:11:01
	pan18-baseline	0.584	00:01:18
7	miller18	0.582	00:30:58
8	schaetti18	0.387	01:17:57
9	gagala18	0.267	01:37:56
10	tabealhoje18	0.028	02:19:14

Table 3. Results regarding the **evaluation** corpus $\mathcal{C}_{Eval}$. Results are adapted from the TIRA evaluation platform [25] (http://www.tira.io).

seen from Table 3, COBAA has been ranked third with results similar to the top performing participants. Furthermore, when comparing Table 2 to Table 3 we can see that the results in terms of Mean Macro-F_1 are quite similar to each other. From this we can infer that COBAA or more precisely, the underlying compression model in combination with the CBC measure, is able to generalize across both corpora. Unfortunately, at the

[6] Available under https://pan.webis.de/clef18/pan18-web/author-identification.html

time this paper was written, the test corpus was not publicly released such that we could not analyze the results on a fine grained level of detail.

4 Conclusion and Future Work

We presented our lightweight approach COBAA, which can be used to solve cross domain authorship attribution problems such as genre or topic. COBAA delegates the feature engineering procedure to a compression algorithm (PPMd) and, therefore, does not involve explicitly defined features. Furthermore, the method does not make use of thresholds or any other trainable (hyper-)parameters. As a consequence, COBAA can be used in realistic scenarios, where training data is not available. Our method has shown its potential at the PAN 2018 Author Identification shared task, where it has been ranked third among 11 participating AA approaches. Aside from the official **test** corpus used in this competition, COBAA was also applied on the given **training** dataset, which we considered as an additional evaluation corpus. Here, we were able to outperform the baseline (a character n-gram-based SVM) that was also used at the PAN 2018 competition. We provided all necessary details to reimplement our approach, which essentially consists only of two components (compression algorithm and a similarity measure).

The characterization of COBAA being independent of a training procedure is also a clear disadvantage of the method, as further optimizations are not possible, at least in its current form. To counteract this, several directions for future work can be considered. One question we wish to answer is, if the attribution results can be improved by applying an ensemble of **several** compression algorithms, instead to rely on only **one**. Another question is, in which way COBAA can be modified to take sophisticated linguistic features such as part-of-speech, chunk or relation tags into account. Also, we would like to investigate the question if instead modifying the method's internals, it would make more sense to transform the method's input texts, in order to achieve better attribution results. Possible text transformations are for instance: lowercasing, elimination of punctuation marks or the more advanced technique "text distortion" ([28]) such that the question is, if COBAA can take advantage from the modified texts. Another direction for future work is to gain a deeper understanding regarding the representation of the compressed texts. Up until now, we do not understand if the compression based model is in fact able to model something we refer to as "writing style".

5 Acknowledgments

This work was supported by the German Federal Ministry of Education and Research (BMBF) under the project "DORIAN" (Scrutinise and thwart disinformation).

References

1. Brennan, M.R., Greenstadt, R.: Practical Attacks Against Authorship Recognition Techniques. In: Haigh, K.Z., Rychtyckyj, N. (eds.) IAAI. AAAI (2009), http://dblp.uni-trier.de/db/conf/iaai/iaai2009.html#BrennanG09 ↑ 1

2. Cerra, D., Datcu, M., Reinartz, P.: Authorship Analysis Based on Data Compression. Pattern Recognition Letters 42, 79 – 84 (2014), http://www.sciencedirect.com/science/article/pii/S0167865514000336 ↑2
3. Chandrasekaran, R., Manimannan, G.: Use of Generalized Regression Neural Network in Authorship Attribution. International Journal of Computer Applications 62(4), 7–10 (January 2013) ↑2
4. Cleary, J., Witten, I.: Data Compression Using Adaptive Coding and Partial String Matching. IEEE Transactions on Communications 32(4), 396–402 (Apr 1984) ↑2, ↑3
5. Cleary, J.G., Teahan, W.J., Witten, I.H.: Unbounded length contexts for ppm. In: Data Compression Conference, 1995. DCC '95. Proceedings. pp. 52–61 (Mar 1995) ↑3
6. El-Yaniv, R., Etzion-Rosenberg, N.: Hierarchical Multiclass Decompositions with Application to Authorship Determination. CoRR abs/1010.2102 (2010), http://arxiv.org/abs/1010.2102 ↑1
7. Gamon, M.: Linguistic Correlates of Style: Authorship Classification with Deep Linguistic Analysis Features. In: Proceedings of Coling 2004. pp. 611–617. International Conference on Computational Linguistics (August 2004), http://aclweb.org/anthology/C/C04/C04-1088.pdf ↑1
8. Goldstein-Stewart, J., Winder, R., Sabin, R.E.: Person Identification from Text and Speech Genre Samples. In: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics. pp. 336–344. EACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009), http://dl.acm.org/citation.cfm?id=1609067.1609104 ↑1, ↑2
9. Halvani, O., Graner, L.: Author Clustering based on Compression-based Dissimilarity Scores. In: Cappellato, L., Ferro, N., Goeuriot, L., Mandl, T. (eds.) Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland, September 11-14, 2017. CEUR Workshop Proceedings, vol. 1866. CEUR-WS.org (2017), http://ceur-ws.org/Vol-1866/paper_59.pdf ↑5
10. Halvani, O., Graner, L., Vogel, I.: Authorship verification in the absence of explicit features and thresholds. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) Advances in Information Retrieval. pp. 454–465. Springer International Publishing, Cham (2018) ↑2
11. 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. pp. 54:1–54:10. ARES '17, ACM, New York, NY, USA (2017), http://doi.acm.org/10.1145/3098954.3104050 ↑2, ↑4, ↑5
12. Hurtado, J., Taweewitchakreeya, N., Zhu, X.: Who Wrote this Paper? Learning for Authorship De-Identification using Stylometric Featuress. In: Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration (IEEE IRI 2014). pp. 859–862 (Aug 2014) ↑1, ↑2
13. Kestemont, M., Tschugnall, M., Stamatatos, E., Daelemans, W., Specht, G., Stein, B., Potthast, M.: Overview of the Author Identification Task at PAN-2018: Cross-Domain Authorship Attribution and Style Change Detection. In: Cappellato, L., Ferro, N., Nie, J.Y., Soulier, L. (eds.) Working Notes Papers of the CLEF 2018 Evaluation Labs. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2018) ↑4, ↑6
14. Kjell, B.: Authorship Attribution of Text Samples Using Neural Networks and Bayesian Classifiers. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics. vol. 2, pp. 1660–1664 vol.2 (Oct 1994) ↑2
15. Kjell, B.: Authorship Determination Using Letter Pair Frequency Features with Neural Network Classifiers. Literary and Linguistic Computing 9(2), 119–124 (1994), +http://dx.doi.org/10.1093/llc/9.2.119 ↑2
16. Koppel, M., Schler, J., Argamon, S.: Authorship attribution in the wild. Language Resources and Evaluation 45(1), 83–94 (2011) ↑2

17. Lambers, M., Veenman, C.J.: Forensic Authorship Attribution Using Compression Distances to Prototypes. In: Geradts, Z.J.M.H., Franke, K., Veenman, C.J. (eds.) Computational Forensics, Third International Workshop, IWCF 2009, The Hague, The Netherlands, August 13-14, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5718, pp. 13–24. Springer (2009), https://doi.org/10.1007/978-3-642-03521-0_2 ↑2
18. Marton, Y., Wu, N., Hellerstein, L.: On Compression-Based Text Classification. In: Losada, D.E., Fernández-Luna, J.M. (eds.) Advances in Information Retrieval, 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3408, pp. 300–314. Springer (2005), http://dx.doi.org/10.1007/978-3-540-31865-1_22 ↑2
19. Mendenhall, T.C.: The Characteristic Curves of Composition. Science 9(214), 237–249 (1887), http://www.jstor.org/stable/1764604 ↑1
20. Moffat, A.: Implementing the PPM Data Compression Scheme. IEEE Transactions on Communications 38(11), 1917–1921 (Nov 1990) ↑3
21. Mohsen, A.M., El-Makky, N.M., Ghanem, N.: Author Identification Using Deep Learning. In: 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 898–903 (Dec 2016) ↑1
22. Nagaprasad, S., Reddy, V., Babu, A.: Authorship Attribution based on Data Compression for Telugu Text. International Journal of Computer Applications 110(1), 1–5 (January 2015), full text available ↑2
23. Pavelec, D., Oliveira, L.S., Justino, E., Neto, F.D.N., Batista, L.V.: Compression and stylometry for author identification. In: Proceedings of the 2009 International Joint Conference on Neural Networks. pp. 669–674. IJCNN'09, IEEE Press, Piscataway, NJ, USA (2009), http://dl.acm.org/citation.cfm?id=1704175.1704273 ↑2
24. Potha, N., Stamatatos, E.: A profile-based method for authorship verification. In: Artificial Intelligence: Methods and Applications: 8th Hellenic Conference on AI, SETN 2014, Ioannina, Greece, May 15–17, 2014. Proceedings. pp. 313–326. Springer International Publishing (2014) ↑5
25. Potthast, M., Gollub, T., Rangel, F., Rosso, P., Stamatatos, E., Stein, B.: Improving the Reproducibility of PAN's Shared Tasks: Plagiarism Detection, Author Identification, and Author Profiling. In: Kanoulas, E., Lupu, M., Clough, P., Sanderson, M., Hall, M., Hanbury, A., Toms, E. (eds.) Information Access Evaluation meets Multilinguality, Multimodality, and Visualization. 5th International Conference of the CLEF Initiative (CLEF 14). pp. 268–299. Springer, Berlin Heidelberg New York (Sep 2014) ↑8
26. Raju, N.V.G., Chinta, S.R.: Region based instance document (rid) approach using compression features for authorship attribution. Annals of Data Science (Jan 2018), https://doi.org/10.1007/s40745-018-0145-4 ↑2
27. Sculley, D., Brodley, C.E.: Compression and Machine Learning: A New Perspective on Feature Space Vectors. In: DCC. pp. 332–332. IEEE Computer Society (2006), http://dblp.uni-trier.de/db/conf/dcc/dcc2006.html#SculleyB06 ↑5
28. Stamatatos, E.: Authorship Attribution Using Text Distortion. In: Proceedings of the 15th Conference of the European Chapter of the Association for the Computational Linguistics, EACL 2017, April 3-7, 2017, Valencia, Spain. The Association for Computer Linguistics (2017), http://www.icsd.aegean.gr/lecturers/stamatatos/papers/eacl2017.pdf ↑2, ↑9
29. Veenman, C.J., Li, Z.: Authorship Verification with Compression Features. In: Working Notes for CLEF 2013 Conference , Valencia, Spain, September 23–26, 2013 (2013), http://ceur-ws.org/Vol-1179/CLEF2013wn-PAN-VeenmanEt2013.pdf ↑2
30. Zhang, C., Wu, X., Niu, Z., Ding, W.: Authorship identification from Unstructured Texts. Knowledge-Based Systems 66, 99 – 111 (2014), http://www.sciencedirect.com/science/article/pii/S0950705114001476 ↑2