Linguistic Tools in Musical Stylometry

Kirill Abrosimov[1], Alexander Grebennikov[2, 3], George Tzanetakis[1], and Anna Sidorova[4]

[1] Faculty of Engineering and Computer Science, University of Victoria, Victoria, Canada
[2] Faculty of Philology, St. Petersburg State University, Saint Petersburg, Russia
[3] Foreign Language Training Center, ITMO University, Saint Petersburg, Russia
[4] AI Talent Hub, ITMO University, Saint Petersburg, Russia

Abstract

In this paper, we investigate the applicability of linguistic stylometry methods to authorship attribution in music. We compare the use of delta methods involving the the analysis of token frequencies with static embeddings generated by distributional semantic models (Word2Vec and Doc2Vec) for the stylometry analysis of music using a symbolic representation. For this purpose, a classical music dataset derived from the MusicNet dataset is used. Applying the cosine delta approach, an F1 score of 0.63 was obtained in a classification approach to authorship attribution. Static embeddings achieved an Adjusted Rand Index of 0.42 using a clustering approach. In both cases, pre-processed extracted chords were used as tokens. We hypothesize that the frequency of using certain chords provides sufficient information to achieve reliable results in symbolic music stylometry analysis. Several methods of chord preprocessing were investigated: augmentation, lemmatization, and n-gram forming, with lemmatization being shown to be the most effective. The proposed methodology influenced by linguistics combined with the chord pre-processing methods can also be used for other tasks in symbolic music information retrieval.

Keywords: musical stylometry, symbolic music processing, representations of music, delta methods, static embeddings

1 Introduction

Musical stylometry is a branch of music data mining that deals with the stylistic characteristics of musical pieces belonging to different eras or composers. One of the central tasks in musical stylometry is authorship attribution, based on a feature representation [2]. Originally, stylometry comes from linguistics, where a large number of attribution methods have been developed, ranging from statistical approaches to pattern recognition techniques [4; 6; 10; 20]. Many of these methods have been adapted and used for musical data processing.

Identifying composers relates to several tasks, the main ones being music authorship attribution and composer classification. Conceptually, both of these tasks aim to answer the question: Who is the author of a particular piece of art? However, composer classification focuses on building a system that performs classification accurately rather than providing insights about why the authorship attribution was made. In contrast, attribution of authorship to music focuses more on feature representation and analysis as a whole, highlighting the most informative features known as style markers [2; 3; 18]. Therefore, unsupervised approaches such as clustering and visualization are commonly used in addition to classification in stylometry analysis within the context of music authorship attribution. Moreover, following the stylometric methodology borrowed from

Kirill Abrosimov, Alexander Grebennikov, George Tzanetakis, and Anna Sidorova. “Linguistic Tools in Musical Stylometry.” In: *Computational Humanities Research 2025*, ed. by Taylor Arnold, Margherita Fantoli, and Ruben Ros. Vol. 3. Anthology of Computers and the Humanities. 2025, 607–618. https://doi.org/10.63744/Av1c2rVmcj0N.

linguistics to determine the authorship of a specific work, stylistically similar and representative examples are selected for comparison [6].

In the 21st century, composer classification is more popular among researchers, and they often train machine learning systems on large datasets [8; 14; 29]. Some researchers, using advances in computer vision, have trained systems based on images of musical scores and achieved good classification results [28]. However, from a musical stylometry perspective, there have not been as many studies, but those existing are of high quality and demonstrate diverse feature representations (information theory based features, statistical features, and acoustic features) [2; 3]. So far, a limited number of linguistic methods have been applied to the analysis of music.

Our goal is to focus on well-established methods in linguistics: delta methods involving analysis of token frequencies in texts [1; 5; 11; 12; 13; 16; 25] and static embeddings [15; 19; 21; 23] to demonstrate their applicability to music stylometry and to obtain quantitative results on a constrained but representative dataset.

2 Methodology

In this section, we describe our methodology for the music attribution task, the dataset chosen, the extracted features, and the linguistic methods we have applied to music data. Figure 1 demonstrates a processing pipeline for this task.

2.1 Data

For an experiment in musical attribution, we need a dataset of the music symbolic representation (e.g., MIDI, musicXML, or ABC notation). We used an open source data set MusicNet [27] that consists of 330 freely licensed classical music recordings. Originally, the data set was used to solve the problem of automatic musical transcription [26; 27], but it also contains MIDI files that are of interest to us.

According to the stylometry methodology [20], to solve the attribution task, the samples of potential authors should be chosen from a limited set according to the time period or style of a potential unknown composition. We have decided to analyze romantic-style compositions (quintets and quartets) listed below indicating composition id (there are totally 38 compositions):

1. **Schubert** [Ids: 1727, 1728, 1729, 1730, 1742]: Piano Quintet in A major (2. Andante, 3. Scherzo Presto, 4. Andantino Allegretto, 5. Allegro giusto); String Quintet in C major (3. Scherzo Presto).

2. **Dvorak** [Ids: 1916, 1918, 1919]: String Quartet No 10 in E-flat major (1. Allegro ma non troppo, 3. Romanza, 4. Finale).

3. **Brahms** [Ids: 2138, 2140, 2148, 2149, 2150, 2151]: String Quartet in C minor (2. Romanze Poco adagio, 4. Allegro); Piano Quartet No 1 in G minor (1. Allegro, 2. Intermezzo Allegro ma non troppo, 3. Andante con moto, 4. Rondo alla Zingarese: Presto).

4. **Faure** [Ids: 2166, 2167, 2168, 2169]: Piano Quartet No 2 in G minor (1. Allegro molto moderato, 2. Allegro molto, 3. Adagio non troppo, 4. Allegro molto).

5. **Beethoven** [Ids: 2313, 2314, 2315, 2365, 2366, 2368, 2376, 2379, 2381, 2382, 2383, 2384, 2431, 2432, 2433, 2494, 2497, 2621, 2622]: String Quartet No 15 in A minor (1. Allegro, 2. Allegro ma non tanto, 3. Adagio Molto Adagio; Andante); String Quartet No 12 in E-flat major (1. Maestoso - Allegro, 2. Adagio ma non troppo e molto cantabile, 4. Allegro); String Quartet No 8 in E minor (1. Allegro, 2. Adagio Molto adagio, 4. Presto Finale, Presto, alla breve); String Quartet No 13 in B-flat major (1. Adagio ma non troppo - Allegro, 2. Presto,

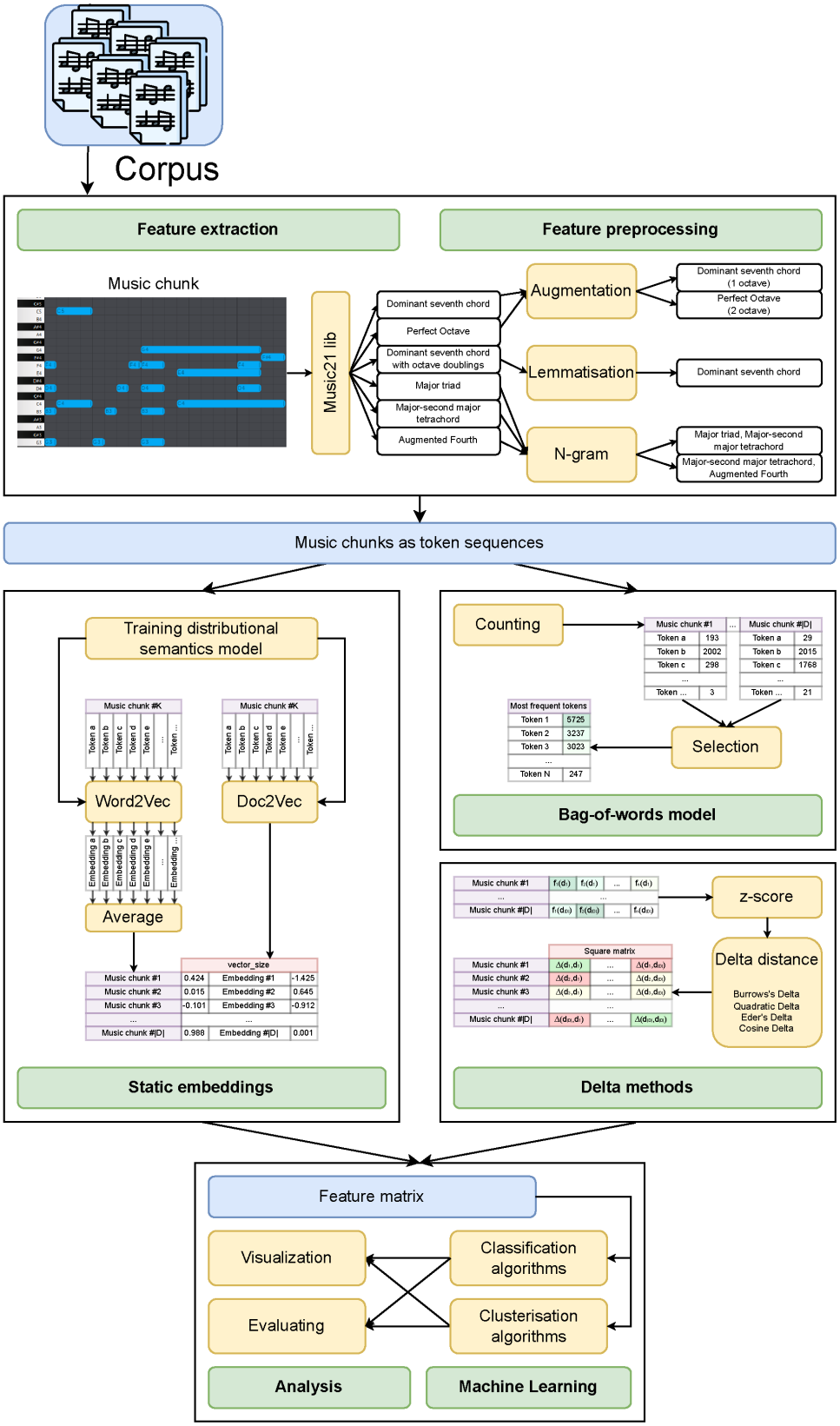

Figure 1: The proposed methodology for authorship attribution in musical stylometry using linguistic approach: extracting and processing chords, applying delta methods or distributional semantics models, analyzing results through solving the tasks of clusterization and classification .

3. Andante con moto, ma non troppo - Poco scherzoso, 4. Allegro Alla danza tedesca - Allegro assai); String Quartet No 16 in F major (1. Allegretto, 2. Vivace, 3. Lento assai - cantante e tranquillo); String Quartet No 11 in F minor (1. Allegro con brio, 4. Larghetto espressivo - Allegretto agitato - Allegro); String Quartet No 7 in F major (1. Allegro, 2. Allegretto vivace e sempre scherzando);

2.2 Token extraction

In this research, we extract musical chords from pieces (chunks) using the music 21 toolkit [7] which supports their automatic extraction from scores/MIDI files. This is accomplished using the *chordify* method which extracts chords for each note by adding all notes that are sounding at the same time. It is important to note that this is a note by note representation that captures melody and surrounding context rather than the more common notion of chord as a harmonic block that spans multiple notes of the melody. This is sometimes referred to as salami slicing and can be visualized as taking vertical columns of a piano roll to form "chords". In the remainder of the paper, this is what is meant by the term chord. Analogously to the approaches used in linguistics, we hypothesize that information regarding only the chords used, without considering their duration, may be sufficient for investigations in musical stylometry. The extracted chords without any processing are referred to as '**Usual**'. We propose the following pre-processing options for these chords: augmentation, lemmatization, and n-gram forming.

1. **Augmentation**: This process involves adding information about the octave in which the chord's bass note is played. In other words, information about the initial sounding octave is added to the chord.

2. **Lemmatization**: Borrowing the term from linguistics, lemmatization refers to reducing a word to its base form. In the context of musicology, we apply this term to transform chords into a narrower set of general chords. This process involves the following transformations: all chords labeled "*<Chord> with octave doublings*" are converted to the standard chord name "*<Chord>*"; specific octave descriptors ("*Perfect Triple-octave*", "*multiple octaves*") become "*Perfect Octave*"; and specialized chords of the form "*enharmonic equivalent to <Chord>*" and "*enharmonic to <Chord>*" are transformed to simply "*<Chord>*".

3. **N-gram forming**: The application of n-gram models from linguistics may also be effective in computational musicology. **N**-gram forming involves combining the nearest **n** chords into a single token with a shift of one.

After the chord processing stage, the result is referred to as a token sequence of the musical chunk.

2.3 Delta Methods

Delta methods using word frequencies are popular in linguistics to identify the authors of texts. In our research, four delta methods are used: the Burrows delta, Eder's delta, Argmon's delta, and the Cosine delta. The algorithm of finding similarities and dissimilarities between documents is the same for each method.

- D is a dataset of documents of various authors, the texts chosen must be related to unknown or test texts (e.g., similar style or time period) based on linguistic stylometry methodology [5].

- Each document $d \in D$ is described as a set of token frequencies $F(d) = (f_1(d), f_2(d), f_3(d), \ldots, f_n(d))$, where f_t is the token frequency t in d, n is the number of the most frequent tokens in the entire dataset $t_i \in D, i \in 1..n$. In addition, the tokens are ordered in descending order based on the total frequencies in the entire dataset.

- To reduce variance of frequencies between tokens (this is particularly important due to Zipf's law [17]) the z-transformation (or z-score) is used to standardize and normalize features:

$$z_i(F_i(d)) = \frac{f_i(d) - \bar{f}_i}{\sigma_{f_i}} \tag{1}$$

  where $\bar{f}_i$ is the average value of the frequency of the token ith in the data set and σ_{f_i} is the standard deviation (therefore, $\bar{z}_i = 0$ and $\sigma_{z_i} = 1$). The effectiveness of normalization and standardization for text attribution has been demonstrated in [13].

- Delta methods are used to evaluate the similarities between two documents. Having calculated the numerical estimate of distance between all documents, the resulting distance square matrix M of dimension $|D|$ can be used to analyze attribution applying unsupervised and supervised methods of machine learning [12].

$$M_{|D|} = \begin{pmatrix} 0 & \Delta_{1,2} & \cdots & \Delta_{1,|D|} \\ \Delta_{2,1} & 0 & \cdots & \Delta_{2,|D|} \\ \vdots & \vdots & \ddots & \vdots \\ \Delta_{|D|,1} & \Delta_{|D|,2} & \cdots & 0 \end{pmatrix}$$

Burrows's delta (or classical delta) [5] is the first method of the family calculating Manhattan distance:

$$\begin{aligned} \Delta_B(d_k, d_l) &= \|z(d_k) - z(d_l)\|_1 \\ &= \sum_{i=1}^{n} |z_i(d_k) - z_i(d_l)| \\ &= \sum_{i=1}^{n} |\frac{f_i(d_k) - f_i(d_l)}{\sigma_{f_i}}| \end{aligned} \tag{2}$$

Argamon's delta (or quadratic delta) [1] calculates Euclidean distance:

$$\begin{aligned} \Delta_A(d_k, d_l) &= \|z(d_k) - z(d_l)\|_2 \\ &= \sum_{i=1}^{n} (z_i(d_k) - z_k(d_l))^2 \end{aligned} \tag{3}$$

Eder's delta [11] additionally increases the weight of the most frequently used tokens:

$$\Delta_E(d_k, d_l) = \sum_{i=1}^{n} (|z_i(d_k) - z_i(d_l)| \times \frac{n - i + 1}{n}) \tag{4}$$

Cosine delta [25] calculates cosine distance:

$$\begin{aligned} \Delta_{\cos}(d_k, d_l) = \alpha &= 1 - \frac{z(d_k) \cdot z(d_l)}{\|z(d_k)\| \, \|z(d_l)\|} \\ &= 1 - \frac{\sum_{i=1}^{n} z_i(d_k) \, z_i(d_l)}{\sqrt{\sum_{i=1}^{n} (z_i(d_k))^2} \, \sqrt{\sum_{i=1}^{n} (z_i(d_l))^2}} \, . \end{aligned} \tag{5}$$

Chords	Delta Method	Best max tokens	ARI	AMI	V1
Usual	Classical	210	0.1340	0.1979	0.3228
	Quadratic	250	0.1462	0.1176	0.2599
	Eder	30	**0.1579**	0.2248	0.3526
	Cosine	70	0.1531	**0.2442**	**0.3773**
Augmented	Classical	600	0.1462	0.1176	0.2599
	Quadratic	30	**0.1773**	0.1531	0.2942
	Eder	100	0.1673	0.1730	0.3132
	Cosine	400	0.1630	**0.2639**	**0.3951**
Lemmatized	Classical	170	0.1979	0.1633	0.3057
	Quadratic	170	0.1897	0.1736	0.3114
	Eder	30	0.2208	0.2731	0.3979
	Cosine	10	**0.2452**	**0.3234**	**0.4402**
2-gram (with lemmatization)	Classical	8	**0.1665**	0.0751	0.2366
	Quadratic	8	0.1654	**0.2509**	**0.3838**
	Eder	6	0.1343	0.2295	0.3667
	Cosine	9	0.0951	0.1711	0.3183
3-gram (with lemmatization)	Classical	2	0.1108	0.1754	**0.3185**
	Quadratic	3	0.0285	0.0720	0.2332
	Eder	3	0.0805	**0.1192**	0.2762
	Cosine	30	**0.1462**	0.1176	0.2599

Table 1: Results of clustering experiments using various preprocesed chords and different delta methods.

2.4 Static embeddings

The second linguistic approach used in the research is the creation of static embeddings using Word2Vec [23] and Doc2Vec [19] techniques. Embeddings are vectors in a vector space, which are created by training systems based on the distributional hypothesis [15].

The following algorithms are used to generate static embeddings:

1. **Continuous Bag-of-Words (CBOW)**: it predicts the current word based on the context (using tokens as the context in indicated window size).
2. **Skip-gram**: it predicts the context tokens (surrounding words) based on a given token.
3. **Paragraph Vector Distributed Memory (PV DM)**: this method is similar to the CBOW algorithm. However, it also encodes information about the paragraph (document) into a separate vector. This document vector is then used in the process of predicting the target token. As a result, not only word embeddings but also trained document embeddings are generated.
4. **Paragraph Vector Distributed Bag-of-Words (PV DBOW)**: when training with a Semi-Supervised approach, the context is also selected within a defined window. However, instead of predicting context tokens based on the central token (as in the Skip Gram algorithm), the prediction is based on a learnable document vector.

A key feature is that these algorithms create fixed embeddings (static embeddings). These embeddings are trained within a specific window size in a vector space of a defined dimension (embedding size). Importantly, these embeddings are not changed depending on the context when

Model	Chords	Window Size	Emb Size	ARI	AMI	V1
Word2Vec: CBOW	Usual	9	100	0.0827	0.1726	0.3197
	Lemmatized	5	20	0.1810	0.1942	0.3335
Word2Vec: Skip-Gram	Usual	10	180	0.0908	0.1736	0.3222
	Lemmatized	7	300	0.2181	0.2309	0.3632
Doc2Vec: PV DBOW	Usual	2	10	0.1781	0.1445	0.2977
	Lemmatized	3	30	0.2572	0.1366	0.2789
	Lemmatized	7	5	0.3017	0.2625	0.3744
Doc2Vec: PV DM	Usual	10	30	0.2031	0.2295	0.3604
	Usual	9	10	0.3002	0.2533	0.3806
	Lemmatized	21	10	0.2726	**0.3874**	**0.4943**
	Lemmatized	15	60	**0.4219**	0.3456	0.4632

Table 2: Results of clustering experiments using semi-supervised algorithms to build static embeddings for stylometry analysis.

the vectors are received. The main advantage of static embeddings is that we can train systems on a limited dataset. The resulting vectors will then contain information only from the sequences of tokens provided. This is exactly what we need for stylometry analysis. The system ”absorbs” information from the provided examples and the ones that have to be analyzed in terms of their authorship. This is not possible with contextualized embeddings, such as those generated by models like ELMO [24] or BERT [9], because these larger models require more data.

Unlike Doc2Vec algorithms, which are also trained to represent information about the paragraph, Word2Vec algorithms only produce static embeddings of the tokens themselves. Therefore, in this research, we use averaging of token embeddings within chunks. In this way, each musical chunk, considered as a sequence of tokens, is represented as a vector of embedding size.

2.5 Unsupervised and supervised learning

Once delta methods have been applied, the features are the vectors distances from one chunk to all the others. After training Word2Vec or Doc2Vec models, the features are the resulting static embeddings. Using the feature matrix, we then apply clustering or classification algorithms.

In stylometry analysis, the Ward clustering algorithm is often used. A popular metric for cluster analysis in stylometry is the Adjusted Rand Index (ARI), with the Adjusted Mutual Information (AMI) and V-measure (V1) as additional metrics. ARI is a robust metric for clustering evaluation, which accounts for chance agreement and provides a normalized score between -1 and 1 to quantify the similarity between predicted and ground-truth cluster assignments. Its adjustment for chance makes it superior to the raw Rand index when comparing clustering solutions as it penalizes random groupings and provides a more accurate reflection of clustering quality.

For classification algorithms, preference is given to classic machine learning classification algorithms due to the limited amount of data. Therefore, we use a Leave-One-Out strategy to analyze the results. The quality metrics are accuracy (ACC), precision-macro (P), recall-macro (R), and f1-macro (F1).

3 Experiments and results

Our research involved three main experiments: the application of delta methods and clustering, the application of static embeddings and clustering, and the application of both types of methods followed by classification.

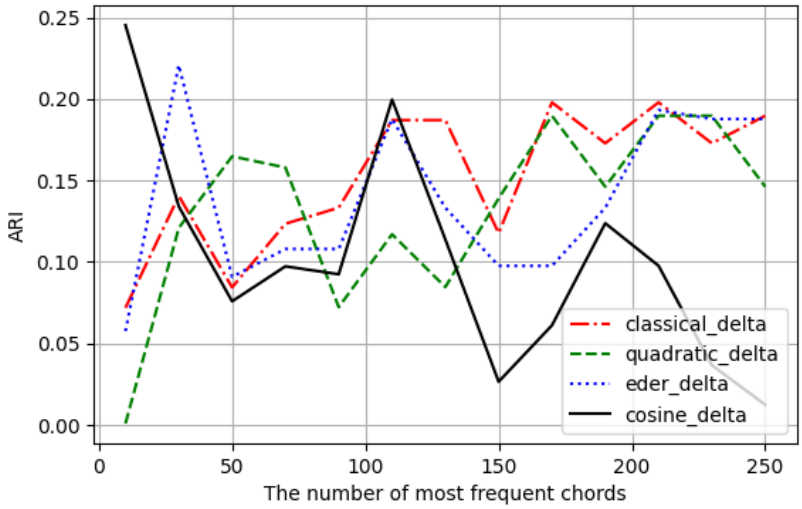

Figure 2: Results of delta methods application using lemmatized chords.

3.1 Delta Methods and clusterization

The first experiment investigates the applicability of delta methods and cluster analysis. Then, the proposed chord preprocessing algorithms (augmentation, lemmatization, and n-gram forming) are evaluated.

Table 1 shows the results of the experiment. The column ”Best max tokens” indicates the optimal hyperparameter value (the maximum number of the most frequent tokens in the dataset) for the system with the best performance metrics. Lemmatization proves to be the most effective preprocessing technique: with its use, any delta method shows higher metrics. The improvement is up to 60% in the main metric, ARI, compared to using tokens (chords) without preprocessing, and 17% in the V1 metric. However, both augmentation and 2-gram preprocessing also improved the quality, but not as noticeably as with lemmatization. Additionaly, forming 3-grams or higher did not result in substantial quality improvements.

Cosine delta, which provides the best results (ARI=0.25, AMI=0.32, V1=0.44), is proved to be the most successful method. Figure 2 shows that the selection of the optimal maximum number of tokens is complex and depends on the choice of the delta method. Generally, it requires further research. This contrasts with linguistics, where there is a clear pattern: the more tokens used, the higher the clustering quality [13]. This difference can be explained by the fact that there are fewer tokens in music data processing (around 300 usual chords and only 250 lemmatized ones). In addition, some tokens may appear only once in a single chunk.

3.2 Static embeddings and clusterization

In this experiment, we used only usual and lemmatized chords. This decision is due to the fact that n-gram models and augmented chords did not provide a considerable improvement in performance. Before generating static embeddings, the model (Word2Vec or Doc2Vec) is trained using one of the available algorithms. The hyperparameters ”*number of epochs*” and ”*minimum token frequency*” were fixed at 200 and 1, respectively. The first parameter was chosen empirically, based on the size of the training data. The second parameter is set to 1 because the total number of tokens used for the research is small. Experiments were conducted with the most important hyperparameters: ”*window size*” and ”*embedding size*”. It is worth noting that the models were trained on the entire

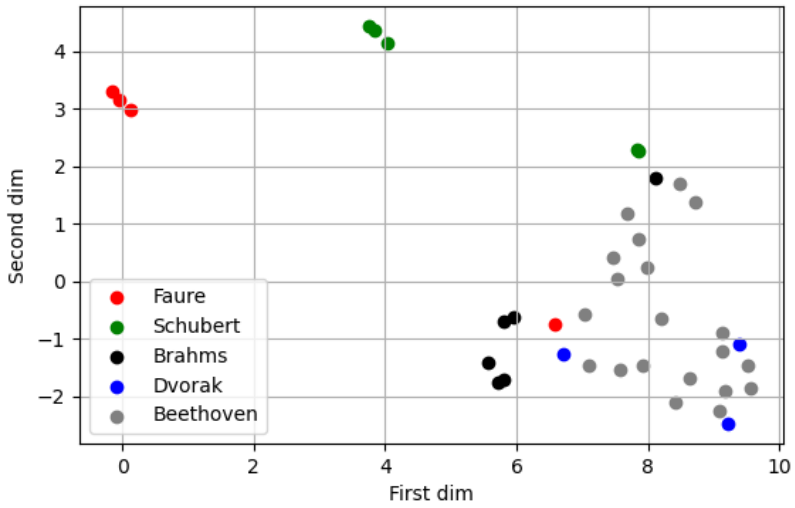

Figure 3: UMAP visualization of embedding space for Doc2Vec model (PV DM algorithm with window size = 21 and embedding space = 10).

Features	Params	Classifier	ACC	P	R	F1
Classical Delta	170	SVM	0.6579	0.2867	0.4667	0.4954
	170	Random Forest	0.6842	0.5741	0.4700	0.4950
Quadratic Delta	170	SVM	0.5789	**0.7214**	0.3833	0.4464
	170	Logistic Regression	0.6579	0.6407	0.5833	0.6084
Eder Delta	30	Logistic Regression	0.6316	0.6111	0.6067	0.6006
	170	SVM	0.6579	0.6027	0.4967	0.5188
Cosine Delta	10	Logistic Regression	0.6842	0.6817	**0.6167**	**0.6269**
	100	SVM	0.6053	0.5567	0.5167	0.5344
Word2Vec: CBOW	(15, 60)	Logistic Regression	0.5526	0.5485	0.5200	0.5281
	(5, 20)	Gradient Boosting	0.6316	0.5055	0.4633	0.4736
Word2Vec: Skip-Gram	(5, 90)	Logistic Regression	0.4474	0.5393	0.5333	0.4754
	(5, 90)	Random Forest	0.6053	0.5200	0.4000	0.4273
Doc2Vec: PV DBOW	(21, 10)	SVM	0.5526	0.5795	0.5133	0.5176
	(21, 10)	Logistic Regression	0.5526	0.4716	0.4667	0.4650
Doc2Vec: PV DM	(21, 10)	Random Forest	**0.7105**	0.5330	0.5033	0.5063
	(7, 20)	Gradient Boosting	0.6316	0.5395	0.5567	0.5436

Table 3: Results of classification experiments with Leave-One-Out strategy for various features using lemmatized chords. For delta features, the maximum number of the most frequent tokens is indicated in the column ”Params”; for distributional semantics models - (window size, embedding size).

dataset, regardless of the potential for musical pieces we intended to analyze. This does not result in data leakage, as these distributional semantic models do not require class labels for training. Therefore, these models are only used to encode information within a specific embedding space.

Static embeddings provided better results than delta methods. Table 2 demonstrates the results of clustering. The PV-DM algorithm performed particularly well, achieving a result of 72% higher for ARI and 12% higher for the V1 score than the best result obtained using delta methods. Similarly to the previous experiment, the results based on lemmatized chords are superior to those based on usual chords.

Figure 3 shows a visualization of the embedding space using the UMAP algorithm [22]. We can see that most of the musical pieces by Faure and Schubert are identified satisfactorily. However, Dvorak's pieces are confused with Beethoven's ones.

3.3 Classification approach to musical stylometry

The classification task, when applied to musical data with Composer labels, is often referred to as Composer Classification. However, in the context of stylometry research, a classification approach is frequently used with clustering. This combination serves to further validate the conclusions, stylistic markers and features that have been identified.

As lemmatized chords were shown to produce the best results, only they are are used for the classification approach.

Due to the extremely limited amount of data available for the classification, we decided to use a Leave-One-Out strategy. This means that the classifier is trained on $(n-1)$ instances and predicts the class of the remaining single instance. After all iterations are complete, performance metrics (accuracy, macro precision, macro recall, and macro F1-score) are calculated.

Since the primary goal is not to build the best possible classifier (as it would be in a Composer Classification task), the hyperparameters of the classifiers are not tuned. Instead, only the default parameter settings are used.

The results are demonstrated in Table 3. Due to space constraints, we only show the results of the two top performing classifiers we experimented with. Despite distributional semantic models having achieved the best metrics' values in clustering, cosine delta in combination with logistic regression (using balanced class weights) provided the best F1-score (0.63) and recall (0.62) in classification. Quadratic delta with an SVM (using an RBF kernel and balanced class weights) had the highest precision (0.72). Only in terms of accuracy did static embeddings combined with Random Forest achieve the best result (accuracy = 0.71). However, in a classification task with imbalanced classes, the F1-score is the most informative metric.

4 Conclusion

The results of this study demonstrate the applicability of linguistic approaches to musical stylometry, specifically delta methods and static embeddings. Similar to linguistics, these methods allow performing stylometry analysis with limited datasets, while still achieving reasonably high results. The proposed pre-processing algorithms for chords performed well, particularly the application of lemmatization to musical chords. However, future research should consider encoding rhythmic information within the tokens, which could improve the results of authorship attribution.

Static embeddings and distance matrices obtained using delta methods have been shown to be effective for solving classification tasks, among others.

References

[1] Argamon, Shlomo. “Interpreting Burrows’s Delta: Geometric and probabilistic foundations”. In: *Literary and linguistic computing 23*, no. 2 (2008), pp. 131–147.

[2] Backer, Eric and Kranenburg, Peter van. “On musical stylometry—a pattern recognition approach”. In: *Pattern Recognition Letters 26*, no. 3 (2005), pp. 299–309.

[3] Brinkman, Andrew, Shanahan, Daniel, and Sapp, Craig. “Musical stylometry, machine learning and attribution studies: A semi-supervised approach to the works of Josquin”. In: *Proc. of the biennial int. conf. on music perception and cognition*. 2016, pp. 91–97.

[4] Burrows, John. “All the way through: testing for authorship in different frequency strata”. In: *Literary and Linguistic Computing 22*, no. 1 (2006), pp. 27–47.

[5] Burrows, John. “Delta: a measure of stylistic difference and a guide to likely authorship”. In: *Literary and linguistic computing 17*, no. 3 (2002), pp. 267–287.

[6] Craig, Hugh and Kinney, Arthur F. *Shakespeare, computers, and the mystery of authorship*. Cambridge University Press, 2009.

[7] Cuthbert, Michael Scott and Ariza, Christopher. “music21: A toolkit for computer-aided musicology and symbolic music data”. In: (2010).

[8] Deepaisarn, Somrudee, Chokphantavee, Sirawit, Chokphantavee, Sorawit, Prathipasen, Phuriphan, Buaruk, Suphachok, and Sornlertlamvanich, Virach. “NLP-based music processing for composer classification”. In: *Scientific Reports 13*, no. 1 (2023), p. 13228.

[9] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[10] Eder, Maciej. “Taking stylometry to the limits: Benchmark study on 5,281 texts from Patrologia Latina”. In: *Digital humanities 2015: conference abstracts*. 2015, pp. 1919–1924.

[11] Eder, Maciej, Rybicki, Jan, and Kestemont, Mike. “Stylometry with R: A package for computational text analysis. R Journal, 8 (1), 107–121”. 2016.

[12] Evert, Stefan, Proisl, Thomas, Jannidis, Fotis, Reger, Isabella, Pielström, Steffen, Schöch, Christof, and Vitt, Thorsten. “Understanding and explaining Delta measures for authorship attribution”. In: *Digital Scholarship in the Humanities 32*, no. suppl_2 (2017), pp. ii4–ii16.

[13] Evert, Stefan, Proisl, Thomas, Vitt, Thorsten, Schöch, Christof, Jannidis, Fotis, and Pielström, Steffen. “Towards a better understanding of Burrows’s Delta in literary authorship attribution”. In: *Proceedings of the Fourth Workshop on Computational Linguistics for Literature*. 2015, pp. 79–88.

[14] Hajj, Nadine, Filo, Maurice, and Awad, Mariette. “Automated composer recognition for multi-voice piano compositions using rhythmic features, n-grams and modified cortical algorithms”. In: *Complex & Intelligent Systems 4* (2018), pp. 55–65.

[15] Harris, Zellig S. “Distributional structure”. In: *Word 10*, no. 2-3 (1954), pp. 146–162.

[16] Jannidis, Fotis, Pielström, Steffen, Schöch, Christof, and Vitt, Thorsten. “Improving Burrows’ Delta. An empirical evaluation of text distance measures”. In: *Digital Humanities Conference*. Vol. 11. 2015, p. 10.

[17] Kingsley Zipf, George. *Selected studies of the principle of relative frequency in language*. Harvard university press, 1932.

[18] Kroonenberg, Pieter M and Kroonenberg, Pieter M. “Musical stylometry: Characterisation of music”. In: *Multivariate Humanities* (2021), pp. 347–370.

[19] Le, Quoc and Mikolov, Tomas. “Distributed representations of sentences and documents”. In: *International conference on machine learning*. PMLR. 2014, pp. 1188–1196.

[20] Love, Harold. *Attributing authorship: An introduction*. Cambridge University Press, 2002.

[21] Markov, Ilia, Gómez-Adorno, Helena, Posadas-Durán, Juan-Pablo, Sidorov, Grigori, and Gelbukh, Alexander. “Author profiling with doc2vec neural network-based title embeddings”. In: *Advances in Soft Computing: 15th Mexican International Conference on Artificial Intelligence, MICAI 2016, Cancún, Mexico, October 23–28, 2016, Proceedings, Part II 15*. Springer. 2017, pp. 117–131.

[22] McInnes, Leland, Healy, John, and Melville, James. “Umap: Uniform manifold approximation and projection for dimension reduction”. In: *arXiv preprint arXiv:1802.03426* (2018).

[23] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

[24] Peters, Matthew E., Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, and Zettlemoyer, Luke. “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, ed. by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 2227–2237. DOI: 10.18653/v1/N18-1202.

[25] Smith, Peter WH and Aldridge, William. “Improving authorship attribution: optimizing Burrows’ Delta method”. In: *Journal of Quantitative Linguistics 18*, no. 1 (2011), pp. 63–88.

[26] Thickstun, John, Harchaoui, Zaid, Foster, Dean P., and Kakade, Sham M. “Invariances and Data Augmentation for Supervised Music Transcription”. In: *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 2018.

[27] Thickstun, John, Harchaoui, Zaid, and Kakade, Sham M. “Learning Features of Music from Scratch”. In: *International Conference on Learning Representations (ICLR)*. 2017.

[28] Walwadkar, Dnyanesh, Shatri, Elona, Timms, Benjamin, Fazekas, György, et al. “CompldNet: Sheet music composer identification using deep neural network”. In: *Proceedings of the 4th International Workshop on Reading Music Systems*. 2022, pp. 9–14.

[29] Wołkowicz, Jacek, Kulka, Zbigniew, and Kešelj, Vlado. “N-gram-based approach to composer recognition”. In: *Archives of Acoustics 33*, no. 1 (2008), pp. 43–55.