Residualized Similarity Prediction for Faithfully Explainable Authorship Verification

Anonymous ACL submission

Abstract

Responsible use of *authorship verification (AV)* systems not only requires high accuracies but also *interpretable* solutions. More importantly, for systems to be used to make decisions with real-world consequences requires faithfulness in a model's prediction. Neural methods achieve high accuracies, but their representations lack direct interpretability. Furthermore, LLM predictions cannot be explained faithfully – if there is an explanation given for a prediction, it doesn't represent the reasoning process behind the model's prediction. In this paper, we introduce **residualized similarity prediction** (RSP), a novel method of supplementing systems using interpretable features with a neural network to improve their performance while maintaining interpretability. The key idea is to use the neural network to predict a *similarity residual*, i.e. the error in the similarity predicted by the interpretable system. Our evaluation across four datasets shows that not only can we match the performance of state-of-the-art authorship verification models, but we can show how and to what degree the final prediction is faithful and interpretable.

1 Introduction

Authorship verification (AV) is a task with many critical applications such as plagiarism detection, forensic linguistics, and literary analysis. In these authorship verification applications, the value of a model lies not only in its prediction accuracy but also in its ability to explain the basis for its prediction. It is in the nature of these applications that the users require *interpretable* solutions, ones where the representations used by the system for verification are simple aggregates of relevant indicators that are used by practitioners and readily understood by stakeholders. Furthermore, it is important that these representations can be verified objectively against the texts that are being investigated. For example, a forensic linguist may rely on linguistic indicators to justify authorship verification. In these cases, it is understood that there is a guaranteed assumption of faithfulness; these linguistic indicators accurately explain the model's prediction (Lyu et al., 2024). Furthermore, the forensic linguist needs to be able to explain how their linguistic indicators were derived from the texts, so that others can agree that they are in fact present in the texts and can be used to argue for or against common authorship.

As with many NLP tasks, representations derived from neural language models often achieve better verification performance than interpretable representations do (Devlin, 2018; Vaswani, 2017).

However, neural representations have major limitations in many critical domains because they are not directly interpretable. When attempts are made to interpret predictions such as in Alshomary et al. (2024), the explanations for a model's predictions are not guaranteed to be faithful to how the prediction was made. In this paper, we ask how one can combine the relative strengths of the two methods: the interpretability and faithfulness of linguistic representations and the high performance of neural models.

As the main contribution of the paper, we introduce *residualized similarity prediction* (RSP), which uses the idea of estimating the *residual* of a predictor i.e., the error in a model's prediction. Suppose we start with an interpretable system as the initial similarity estimator. We can then train a neural model as a *residual predictor*, which predicts the error or correction to the interpretable system's similarity score. The final prediction is a simple sum of the the interpretable model's similarity score and the predicted residual, i.e., a similarity adjustment made by the neural model. This combined system can achieve the trade-off we desire: (i) when the interpretable model is likely to be correct, the residual should be low, providing interpretability and faithfulness while remaining accurate, and (ii) when the interpretable model is likely to be incorrect, the residual should provide the necessary correction, improving accuracy but reducing interpretability to a degree proportional to the error. This approach is inspired by the *residualized control* approach (Zamani et al., 2018), which trains a residual model for a regression problem, combining numerous linguistic features with a few interpretable health-relevant attributes to predict community health indicators. We describe our approach in detail in Section 3.

We use Gram2vec (Zeng et al., 2024) as our interpretable feature system, which records normalized frequencies of morphological and syntactic features for input texts. We evaluate our RSP approach by combining Gram2vec with a state-of-the-art AV neural model, LUAR (Rivera-Soto et al., 2021), finding that RSP can match the performance of using LUAR alone, while introducing interpretability and faithfulness (Sections 5 and 6). We make a distinction between two aspects of faithfulness. First, our system's prediction can be explained directly using the underlying features in Gram2vec. Second, these features are directly measurable within a text, i.e. we can explain exactly why a feature in a given text has a certain value. We perform a case study on how our system retains interpretability, measured by an *interpretability confidence (IC)* metric, which indicates the extent to which the interpretable system is used for a given input. Details of this are in Section 6.

2 Related Work

Authorship verification, authorship attribution, and authorship profiling are all part of authorship analysis which has been explored through a wide range of approaches (see surveys El and Kassou (2014); Misini et al. (2022)).

Interpretable Methods Previous stylometric approaches (Stamatatos, 2016) often make use of readily interpretable features to train classifiers. Some examples include lexical features such as vocabulary, lexical patterns (Mendenhall, 1887; van Halteren, 2004), syntactic rules (Varela et al., 2016), and others.

Neural Models Authorship verification has benefited from models built upon RNNs Gupta et al. (2019), CNNs (Hossain et al., 2021), BERT-like architectures (Manolache et al., 2021), and Longformers (Ordoñez et al., 2020; Nguyen et al., 2023). More recently, sentence-transformer based models (Wegmann et al., 2022; Rivera-Soto et al., 2021) have obtained state-of-the-art performance for AV tasks. As we are interested in improving the performance interpretable authorship verification, we focus on these SOTA AV models. In particular, we focus on LUAR (Rivera-Soto et al., 2021).

Our work uses residual error analysis to combine interpretability and neural models' high performance for authorship verification. Similar residual approaches have been used previously for improving performance in health outcome prediction, by combining lexical and health-relevant attributes (Zamani et al., 2018), and in a recent work that combines statistical and neural methods for machine translation (Benko et al., 2024). Other works have focused on generating explanations, often layering other mechanisms on top of interpretable input features (Boenninghoff et al., 2019; Setzu et al., 2024; Theophilo et al., 2022) or doing a post-hoc evaluation on a latent, non-interpretable space (Alshomary et al., 2024). Some recent work also explores prompting large language models to derive interpretable stylometric features for authorship analysis (Hung et al., 2023; Patel et al., 2023). However, these features are not measurable in a text as the approaches rely on LLMs to generate the features, and the generations do not represent the reasoning process behind attributing a set of features to a text.

3 Residualized Similarity Prediction

The key idea in **residualized similarity prediction** (RSP) is to train a neural model to predict the residual similarity, i.e., the difference between the cosine similarity obtained from the interpretable system and the ground truth. Per each train/dev/test set, we first generate interpretable feature vectors for each document using Gram2vec. Next, to account for difference in variance, the feature vectors are standardized (z-scored) per feature against their respective dataset. Finally, the cosine similarity is calculated between pairs of vectorized documents. The ground truth label is 1 for a pair of documents written by the same author and -1 otherwise. RSP is trained to predict $y - \text{sim}(f(d_1), f(d_2))$, where y is the gold label, sim represents the cosine similarity between the pair of vectorized documents, d_1 and d_2 are the two documents, and f is the Gram2vec vector function. We will call this the *ground truth residual*.

Figure 1 illustrates the specifics of training the

RSP model. The process of training RSP begins with pairs of documents. These are vectorized both by the interpretable system and by the neural model we are fine-tuning, giving us four embeddings. Next, an attention layer is placed over all four embedding, in order for RSP to learn how much to weigh the interpretable features and the neural embeddings when making the residual prediction. Note that this step is only for the training, and interpretability remains simple for the Gram2vec features during inference.

We experimented with some alternatives: earlier attempts included passing only the neural embeddings into the regression head as well as directly appending the interpretable feature vectors to the neural embeddings before passing into the regression head to predict the residual. The former was done to try to capture the power of sequence classification using RoBERTa (Liu, 2019), and the latter was the first attempt to incorporate signal directly from the interpretable system into the training of RSP. However, neither approach was able to match the performance of the contrastive-loss fine-tuned neural model, LUAR, detailed in section 3.1.

Our evaluation tests how the **residualized similarity prediction** method fares against the performance of the two methods it combines: a system using only interpretable features, and a neural model fine-tuned on the target datasets. For the neural model baseline, the neural model on each dataset using a contrastive learning objective (Khosla et al., 2020). We evaluate the systems' performance based on the receiver-operating characteristic area under curve (AUC), as it is a way to measure performance of models that is threshold independent.

3.1 Methods

Gram2vec System: We use Gram2vec to derive interpretable feature vectors from texts. These vectors comprise z-scored relative frequencies of various grammatical features of documents. These vectors are standardized against their respective corpus, e.g., Reddit vectors get standardized against all the other vectors in the Reddit dataset, Amazon vectors get standardized against all the other vectors in the Amazon dataset, and so on. Table 1 shows each feature type and its respective count. The only difference between the Russian and English versions are the types of syntactic constructions that are being searched for. Note that Gram2vec does not use open-class lexical features, and therefore does not model content at all.

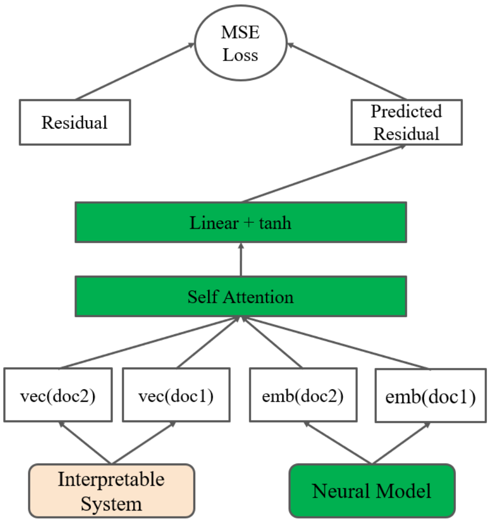


Figure 1: **Residualized Similarity Architecture.** To incorporate signal from the interpretable feature vectors, we add an attention layer over both the interpretable feature vectors as well as the neural embeddings from the model we're fine-tuning. Boxes colored in green indicate that they're updated during training.

Feature type	Count
Punctuation marks	19
Emojis	10
POS Unigrams	18
POS Bigrams	324
Morphology tags	46
Dependency labels	45
Syntactic Constructions	10
Function words	145

Table 1: Counts of different Gram2vec feature categories.

We then compute cosine similarity between the two vectors. If the cosine similarity exceeds a specific threshold (set to 0.5 for analysis in Section 7), we label the input pair as being from the "same author"; otherwise, we label them as being "from different authors".

Contrastive-loss Fine-tuned Neural Model: We focus on LUAR and LUAR-RU, used for English and Russian text respectively. We choose LUAR as it is state-of-the-art in the task of authorship verification (Rivera-Soto et al., 2021), and we also use the Russian version to demonstrate effectiveness across multiple languages. We fine-tune these models in a Siamese network using a contrastive loss function as the training objective.

This approach is similar to SBERT (Reimers and Gurevych, 2019), but we use the architecture to learn document-level, as opposed to sentence-level, semantic embeddings.

Residualized Similarity Prediction: We fine-tune LUAR and LUAR-RU with an attention layer over the interpretable as well as the neural embeddings, with the labels being the ground truth residuals from the training set using Gram2vec similarities. The training process is as follows.
Definitions:

- Let $d1, d2$ = document 1, document 2
- Let y = gold label (1 if same author, -1 if different author)
- Let f = Gram2vec vectorizer
- Let **sim** = cosine similarity function
- Let $y - \textbf{sim}(f(d1), f(d2))$ = the ground truth residual
- Let **res_pred** = predicted residual
- Then **final_score** = ($\textbf{sim}(f(d1), f(d2)) + \textbf{res\_pred}$)
- Let t = threshold for cosine similarity, set to 0.5

Training Process:

- For each document pair i in the training batch:
    - Obtain **res_pred**
    - Calculate MSE Loss: $\frac{1}{n}\sum_{i=1}^{n}(res\_pred_i - res\_actual_i)^2$
    - Update model parameters to minimize MSE

Inference:

- For a new document pair:
    - If **final_score** $> t$: Predict same author
    - Otherwise: Predict different author

Training Details: All neural models and RSP are trained using LoRA (Hu et al., 2021), which reduces the number of trainable parameters and memory requirements. We observe that using LoRA also yields better performance overall for all models as compared to a full fine-tuning. For evaluation of system performance, we use receiver-operating curve area under curve (AUC), which doesn't require tuning of a threshold. Additional training details are in Appendix A.

4 Data

We train and evaluate our **residualized similarity prediction** system on four datasets covering diverse genres. We choose the first three as they are the datasets used by Rivera-Soto et al. (2021) from the original training of LUAR, and we include the Russian dataset Pikabu to evaluate our method on another language as we had access to a Russian version of LUAR.

In order to train both RSP and the contrastive-loss fine-tuned baseline, we require the data to be in a paired format: {Document 1, Document 2, Same/Different label}. The full details of training RSP are provided in section 3. For the contrastive-loss fine-tuned baseline, the aim is to push pairs of documents by the same author together, and to push pairs of documents by different authors apart.

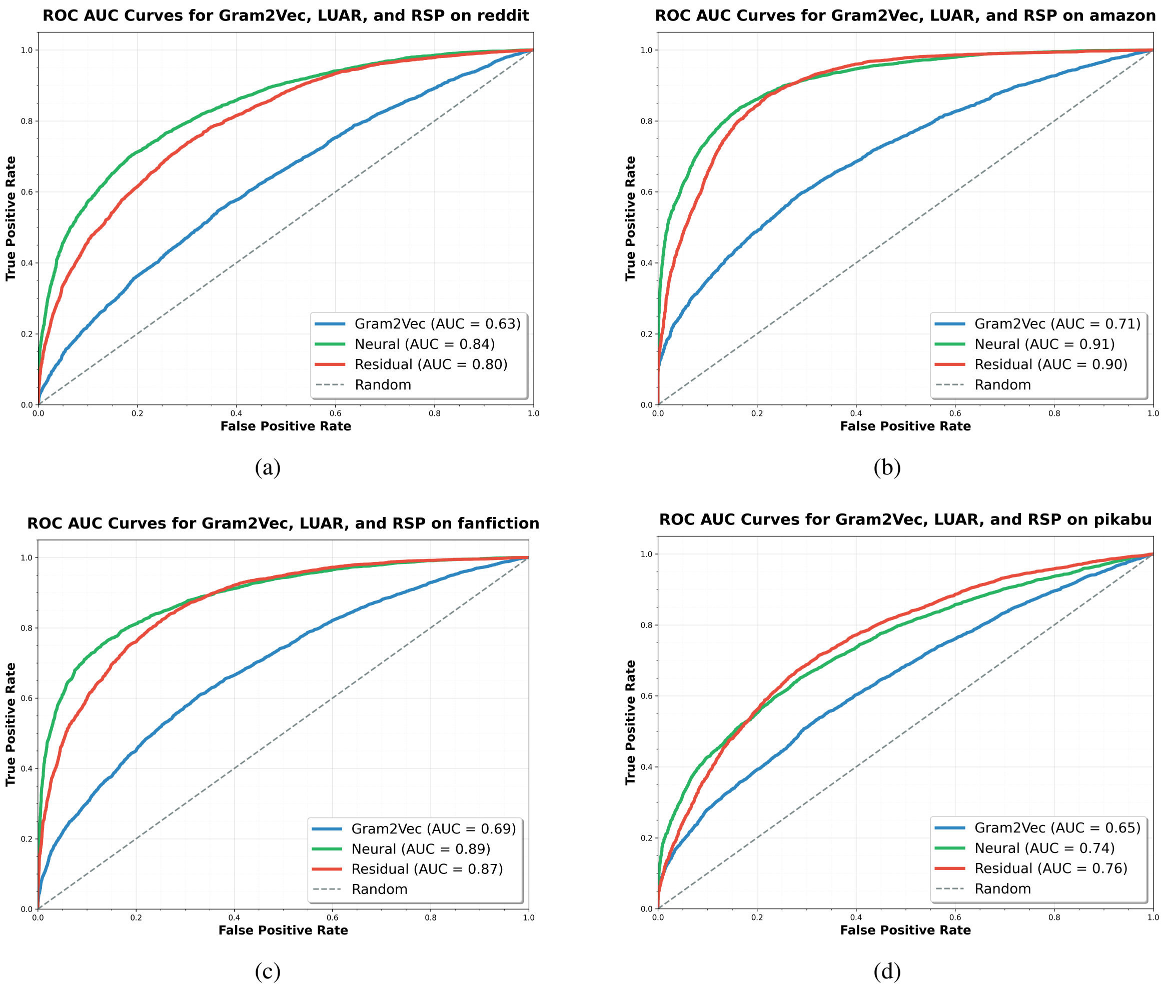
Reddit Comments We use a dataset of Reddit comments from 100 active subreddits created by ConvoKit (Chang et al., 2020). We use a version preprocessed by (Wegmann et al., 2022), as it has invalid comments, with invalid comments, comments containing only some sort of white space or deleted comments, removed and is split into train, development, and test sets with non-overlapping authors. We create pairs of comments, label them for author verification, and use the same split of comments as they do. Reddit comments can be naturally very short, so we further filter the comment pairs and keep only comments longer than 20 words.

Amazon Reviews From the Amazon review dataset (Ni et al., 2019), we take reviews from three categories: Office Products; Patio, Lawn and Garden; and Video games. We use a reduced dataset where all items and users have at least 5 reviews, and we keep authors with at least two reviews of 20 or more words. The validation set is split from the training set by taking stories from 1/6 of the authors. Then, we sample same author pairs by randomly choosing an author and two texts written by them. For different author pairs, two authors and one text from each author are randomly chosen.

Fanfiction Stories The fanfiction dataset contains 75,806 stories from 52,601 authors in the training set and 20,695 stories from 14,311 authors in the evaluation set. We use the pre-processing script from LUAR (Rivera-Soto et al., 2021) to split each story into paragraphs since fanfictions can be very long. The process of sampling pairs of reviews is the same as in the Amazon dataset.

Pikabu comments We start with the Pikabu dataset from Ilya Gusev (2024) available on HuggingFace. We drop documents with fewer than 100 characters, and authors with fewer than two documents; we then anonymize the data, redacting credit card numbers, IP addresses, names, and phone numbers.

For all four datasets, we use 50K, 10K, and 10K pairs for the training, validation, and test sets re-



(a)

(b)

(c)

(d)

Figure 2: ROC AUC Curves for Gram2vec, LUAR, and RSP on the (a) Reddit, (b) Amazon, (c) Fanfiction, and (d) Pikabu datasets. We observe performance increases comparing RSP to Gram2vec that range from 11 points on Pikabu to 25 points on Amazon. Notably, RSP also sees a 2 point increase in performance from LUAR on Pikabu.

spectively. The ratio of same to different author pairs is 1:1.

5 Results

Metrics: We evaluate RSP against both Gram2vec and neural models on the receiver-operating curve area-under-curve (AUC), which represents a model's performance across all thresholds. It is calculated by calculating the true positive rate (TPR) and false positive rate (FPR) at every threshold, and graphing TPR over FPR. We use AUC as it is threshold-independent and the data we use is balanced, providing a direct comparison of the various systems. RSP is able to match or just nearly match the performance of LUAR and LUAR-RU on all four datasets, with even a slight increase in the Pikabu dataset. However, we observe a big increase in performance compared to using Gram2vec alone, with the biggest improvement being an increase of 25 points on the Amazon dataset. We present the AUC curves of the three methods we evaluate on all four datasets in Figure 2.

Summary of Results: By using AUC as our metric, we show that system performance for RSP and LUAR are nearly identical for each dataset, with a slight decrease for Reddit, and a slight increase for Pikabu. Gram2vec alone performs consistently lower than both systems, but still above the random baseline. In this section, we show our first claim that RSP is able to match the performance of the state-of-the-art LUAR. In the next, we show that RSP retains a portion of interpretability that Gram2vec offers and quantify the interpretability.

6 Analysis of Interpretability

We have shown that **residualized similarity prediction** is a hybrid system that uses a neural model to correct the error in prediction made by an inter-

Example Pair 1: Different Author

Document 1:
Whirling like a scythe, the saber sliced her upper torso, putting an end to the vengeful Sith. Dropping to her knees again, Jameh crawled to her fallen Master, cradling him in her arms. A new darkness grew in her heart now, one like a cold, lonely mist. Her Master was dying. Just then, footsteps came down the cave passage and Pilae, Obi-Wan, and Anakin entered the grotto just in time to be too late. They stood nearby, dismayed at the sight that met their eyes: a dismembered former Senator, a shorn and wounded Padawan, and a Jedi Master on the verge of death. " Master, please, you can"t leave me. I need you; I"m not ready!"

Document 2:
As the Clan speculated why the rats weren"t attacking, Redfur walked through the camp entrance tentatively, leaving Sootcloud and Brightnose at their original position at one side of the entrance. He scanned the field beyond and was dumbfounded when he didn"t see any rats. As he walked further out with more confidence, he tasted the air and searched for their distinctive scent. Suddenly, with a loud squeak, several of the rats surged forward out of nowhere, or so Redfur thought, and attacked him. He yowled in surprise as some of the rats managed to climb up his leg and cling to his red brown fur, leaving scratches and bites along the way. He pelted back through the entrance and into the clearing. The Clan had been alerted by Redfur"s yowl of surprise, so they had stopped chatting and lowered their bodies into a crouch, getting ready for the rats. But when they saw the four rats clinging to Redfur"s fur, they hissed in astonishment at the size of them.

Gram2vec Cosine Similarity: 0.09, **RSP Predicted Residual**: 0.29, **Final Score**: 0.38
Interpretability Confidence: 0.72 **Flipped**: False

Example Pair 2: Same Author

Document 1:
GET UP! School time!" Sora called from the door. " I"m up!" he hollered back before throwing the cover"s off him. It"s been a week. A week since Roxas started hearing that voice. Throughout that time he had figured out that it was connected to the mirror he had gotten at the same time. "

Document 2:
It was passed down through generations to keep him in the glass." At this he closed the book and plopped on the bed. " What about the rhyme?" Demyx stroked his chin in a pondering position. " It was created to scare children from letting him out. Though the ending part. "" A curse to never be free of. Until this demon admits love" Is exactly what it says.

Gram2vec Cosine Similarity: 0.20, **RSP Predicted Residual**: 0.82, **Final Score**: 1.02
Interpretability Confidence: 0.18 **Flipped**: True

Figure 3: Example Pairs for Case Study. Pair 1 is by two different authors, and Pair 2 is by the same author.

pretable system. In doing so, we can match the performance of a solely neural system, while retaining interpretability. In this section we discuss how to quantify the amount of interpretability a specific result retains. We introduce the notion of *"interpretability confidence"*(INTCONF), which is a way to measure how interpretable a particular prediction of RSP is. We define INTCONF to have two parts, a score, defined as $1 - |\text{predicted residual}|$, and an indicator of whether or not the label was flipped by the predicted residual (1 if flipped, 0 if not). The label is considered flipped if the cosine similarity prediction using Gram2vec is on one side of the cosine similarity threshold (different author if below, and same author if above), and adding the predicted residual from RSP causes the final score to be on the other side of the threshold. We provide an example of this in Section 7.2. Note that we can calculate the IntConf for any specific pair of documents after running RSP.

We emphasize that even in cases where the prediction is flipped after using RSP we can still make use of the underlying interpretable system. We show in section 7 that when the prediction was changed, the underlying interpretable system can help explain why a prediction was made.

7 Case Study of Two Pairs of Documents

We present two cases to illustrate how RSP can give a user insight while performing a specific authorship verification task. We present two pairs of texts, one of which is indeed from the same author, and one of which is not. We set a threshold of 0.5, as a natural midpoint from 0 to 1, suggesting that documents need to be more similar than dissimilar to be considered by the same author. We show how our approach can tell the user which Gram2vec features were used in the determination, and to what extent they determined the confidence of the prediction. Since Gram2vec contains over 600 features, we define a criterion to select features to present to the user, depending on whether a pair of documents are predicted to be by the same or different authors. When a pair of documents is predicted to be written by the same author, we want to maximize the absolute values of the feature values (features that distinguish these documents from the large set of background documents) while making sure the values are similar for both documents. When a pair of documents is predicted to be written by different authors, we simply find the largest magnitudes of differences in the feature values. Thus, for identifying features for same author pairs, we use the following metrics for ordering features, where val_1 represents the feature's score for document 1, and val_2 represents the feature's score for document 2.: $|val\_1| + |val\_2| - |val\_1 - val\_2|$. For ordering features using different author pairs, we use $|val\_1 - val\_2|$. We then choose the top n features; in the examples below, we use $n = 10$.

7.1 Example 1: Different Author Pair

Looking at the first example in Figure 3, based on a threshold of 0.5, we observe that both Gram2vec and RSP predict that these two documents are written by different authors: the gold label for different authors is -1, and we see that in this case, both Gram2vec at 0.09 and RSP at 0.38 agree, indicating that the label is not flipped. Below, in Table 2, we show the top 10 features and their values that were identified using the different author pair metric: $|val\_1 - val\_2|$. We calculate this score for every feature in document 1 and document 2, and sort in descending order the top 10 features. These represent the 10 most differing features in the pair of documents. Looking at the features, we first note several function words which can be found in document 2 but not in document 1; for example, document 2 uses *when* twice in a fairly short text, while document 1 does not use it at all. In contrast, document 1 uses several part-of-speech (POS) bigrams far more frequently than the background corpus, while document 2's distribution of POS bigrams is more standard. A striking example is the bigram adjective-proper noun, which is unusual in general but very frequent in document 1 (*vengeful Sith*, *fallen Master*, *former Senator*, *wounded Padawan*). Finally, we note the high frequency of the indefinite article in document 1: *a scythe*, *a new darkness*, *a cold, lonely mist*, *a dismembered former senator*, *a shorn and wounded Padawan*, *a Jedi Master*. These indefinite noun phrases provide a sense of change (indefinites introduce new discourse objects); in the case of the last three, the author takes on the perspective of three characters. Document 2, in contrast, has few indefinites and the narration centers on entities known to the readers and the characters in the story.

Feature	Score	Doc 1	Doc 2
func_words:further	5.4	-0.1	5.3
pos_bigrams:ADJ PROPN	4.1	3.8	-0.3
pos_bigrams:PUNCT DET	3.8	3.4	-0.4
func_words:through	3.6	-0.3	3.3
pos_bigrams:PART ADJ	3.3	3.1	-0.2
func_words:they	2.9	-0.4	2.5
pos_bigrams:PROPN PUNCT	2.9	2.1	-0.8
morph_tags:Definite=Ind	2.8	2.4	-0.4
pos_bigrams:PUNCT NUM	2.6	2.5	-0.1
func_words:when	2.6	-0.4	2.2

Table 2: Feature scores comparison between Example 1 document pair by different authors.

7.2 Example 2: Same Author Pair

In this case, based on a threshold of 0.5, we observe that Gram2vec predicts that the two documents are written by different authors, and RSP predict that these two documents are written by the same author. The gold label for the same author is 1, and we see that Gram2vec gets the prediction wrong. However, RSP predicts the similarity residual and the final score is right at the gold label for the same author. Even though the label was flipped from Gram2vec to RSP in this case, we observe that there are still a good number of features that are similar between the two documents which we can use in explanation, since they in fact contributed to the final prediction. When identifying similar features in two documents, we use the metric $|val\_1|+|val\_2|-|val\_1-val\_2|$ and take the top 10 features in descending order, shown in Table 3. Thus, these are features which occur in both documents either much more or much less frequently than on average across a background corpus. One example is the bigram preposition-punctuation. In both texts, we find examples: *UP!*, *up!* (in document 1), *out.*, *of.* (in document 2). A preposition at the end of a clause is often discouraged in formal written English. The two documents also use passive voice clauses more frequently than on average (passive voice is generally rare in written English): *it was connected* (document 1), *it was passed down*, *it was created* (document 2). The two documents share a negative value for the punctuation mark comma. Indeed, neither text contains a comma, which in general is a very common punctuation mark.

Feature	Score	Doc 1	Doc 2
pos_bigrams:PREP PUNCT	6.1	4.1	3.0
passive sentence	5.4	2.7	4.6
dep_labels:nsubjpass	4.4	2.2	3.8
pos_bigrams:PREP VERB	4.3	2.9	2.1
dep_labels:auxpass	3.7	1.8	3.3
func_words:from	3.5	2.3	1.8
punctuation:,	3.4	-1.7	-1.7
morph_tags:PunctType=Comm	3.3	-1.6	-1.6
pos_bigrams:DET NOUN	3.2	2.5	1.6
func_words:the	2.9	1.5	1.5

Table 3: Feature scores comparison between Example 2 document pair by the same author.

We note that this paper does not propose an end-to-end explainable system. Instead, we have shown how our RSP system can identify measurable features which it actually used in determining its finding (faithfulness), and it can quantify to what extent these features explain why the system came to its result. An explainable system built on top of our system would require in addition two types of decisions: how do we choose how many and which features to present to the user, and exactly how should the interface look? These are, at base, human-computer interface (HCI) issues: explanations are always for a particular type of user, and need to be tailored to that user. If, for example, our target audience is forensic linguists, then we can assume that they know the meaning of linguistic features and are willing to get to know a more complex interface (which, for example, may allow them to drill down, or to include or exclude certain types of linguistic features). If on the other hand the target audience is crowdsourced workers (because we are doing an evaluation for a paper for a submission to an NLP conference, for example), then of course we cannot assume the users will know the meaning of our features, nor that they will take the time to get to know the capabilities of a more complex interface. We leave this HCI work to a future publication.

8 Conclusion

We introduce **residualized similarity prediction**, a method of improving the performance of an interpretable feature set by training a language model to predict the residual, or difference, between the similarity output from an interpretable system and the ground truth. Using **residualized similarity prediction**, we are able to achieve state-of-the-art performance while maintaining a degree of interpretability.

To measure interpretability, we introduce the **interpretability confidence**, a measure of how interpretable a prediction from our system is. We then do a case study to observe how using RSP, we are able to correct a prediction that was initially incorrect from an interpretable system. In both the case where the prediction was corrected and the case where the prediction from the interpretable system and RSP agreed, we show that there is meaningful interpretability in the features.

We believe this approach to be a promising direction for developing more interpretable and effective NLP systems, bridging the gap between neural methods and interpretable linguistic features while allowing for faithfully explainable systems.

Limitations

We present preliminary results on **residualized similarity prediction** (RSP), a novel method of supplementing systems using interpretable linguistic features with a neural network to improve their performance while maintaining interpretability. In order to get these results, we use a relatively small subset of data from the original datasets we chose.

While we choose a variety of datasets, our experiments are by no means conclusive.

The goal of this work is to improve performance while maintaining interpretability. With this in mind, we developed the **interpretability confidence**, a way to quantify how interpretable predictions from RSP are. Thus, if we find that the majority of residual predictions in fact flip the original prediction or have high magnitudes, then RSP will have less interpretability than desired.

Ethics Statement

The datasets we use are publicly available and are anonymized. Our work improves the interpretability of authorship verification models, allowing for more transparency and easier detection of potential biases and errors in the model.

References

Milad Alshomary, Narutatsu Ri, Marianna Apidianaki, Ajay Patel, Smaranda Muresan, and Kathleen McKeown. 2024. Latent space interpretation for stylistic analysis and explainable authorship attribution. *arXiv preprint arXiv:2409.07072*.

L'ubomír Benko, Dasa Munkova, Michal Munk, Lucia Benkova, and Petr Hajek. 2024. The use of residual analysis to improve the error rate accuracy of machine translation. *Scientific Reports*, 14(1):9293.

Benedikt Boenninghoff, Steffen Hessler, Dorothea Kolossa, and Robert M. Nickel. 2019. Explainable authorship verification in social media via attention-based similarity learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 36–45.

Jonathan P. Chang, Caleb Chiam, Liye Fu, Andrew Wang, Justine Zhang, and Cristian Danescu-Niculescu-Mizil. 2020. ConvoKit: A toolkit for the analysis of conversations. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 57–60, 1st virtual meeting. Association for Computational Linguistics.

Jacob Devlin. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sara El Manar El and Ismail Kassou. 2014. Authorship analysis studies: A survey. *International Journal of Computer Applications*, 86(12).

Shriya TP Gupta, Jajati Keshari Sahoo, and Rajendra Kumar Roul. 2019. Authorship identification using recurrent neural networks. In *Proceedings of the 2019 3rd International Conference on Information System and Data Mining*, ICISDM '19, page 133–137, New York, NY, USA. Association for Computing Machinery.

Md Rajib Hossain, Mohammed Moshiul Hoque, M Ali Akber Dewan, Nazmul Siddique, Md Nazmul Islam, and Iqbal H Sarker. 2021. Authorship classification in a resource constraint language using convolutional neural networks. *IEEE Access*, 9:100319–100338.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Chia-Yu Hung, Zhiqiang Hu, Yujia Hu, and Roy Ka-Wei Lee. 2023. Who wrote it and why? prompting large-language models for authorship verification. *Preprint*, arXiv:2310.08123.

Ilya Gusev. 2024. pikabu (revision 96466c2).

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

I Loshchilov. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2024. Towards faithful model explanation in nlp: A survey. *Computational Linguistics*, pages 1–67.

Andrei Manolache, Florin Brad, Elena Burceanu, Antonio Barbalau, Radu Ionescu, and Marius Popescu. 2021. Transferring bert-like transformers' knowledge for authorship verification. *arXiv preprint arXiv:2112.05125*.

T. C. Mendenhall. 1887. The characteristic curves of composition. *Science*, 9(214):237–249.

Arta Misini, Arbana Kadriu, and Ercan Canhasi. 2022. A survey on authorship analysis tasks and techniques. *SEEU Review*, 17(2):153–167.

Trang Nguyen, Charlie Dagli, Kenneth Alperin, Courtland Vandam, and Elliot Singer. 2023. Improving long-text authorship verification via model selection and data tuning. In *Proceedings of the 7th Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 28–37, Dubrovnik, Croatia. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, Hong

Kong, China. Association for Computational Linguistics.

Juanita Ordoñez, Rafael Rivera Soto, and Barry Y Chen. 2020. Will longformers pan out for authorship verification. *Working Notes of CLEF*.

Ajay Patel, Delip Rao, Ansh Kothary, Kathleen McKeown, and Chris Callison-Burch. 2023. Learning interpretable style embeddings via prompting LLMs. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15270–15290, Singapore. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rafael A. Rivera-Soto, Olivia Elizabeth Miano, Juanita Ordonez, Barry Y. Chen, Aleem Khan, Marcus Bishop, and Nicholas Andrews. 2021. Learning universal authorship representations. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 913–919, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mattia Setzu, Silvia Corbara, Anna Monreale, Alejandro Moreo, and Fabrizio Sebastiani. 2024. Explainable authorship identification in cultural heritage applications. *J. Comput. Cult. Herit.* Just Accepted.

Efstathios Stamatatos. 2016. Authorship verification: A review of recent advances. *Res. Comput. Sci.*, 123:9–25.

Antonio Theophilo, Rafael Padilha, Fernanda A. Andaló, and Anderson Rocha. 2022. Explainable artificial intelligence for authorship attribution on social media. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2909–2913.

Hans van Halteren. 2004. Linguistic profiling for authorship recognition and verification. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 199–206, Barcelona, Spain.

Paulo Varela, Edson Justino, Alceu Britto, and Flávio Bortolozzi. 2016. A computational approach for authorship attribution of literary texts using sintatic features. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 4835–4842. IEEE.

A Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Anna Wegmann, Marijn Schraagen, and Dong Nguyen. 2022. Same author or just same topic? towards content-independent style representations. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 249–268, Dublin, Ireland. Association for Computational Linguistics.

Mohammadzaman Zamani, H. Andrew Schwartz, Veronica Lynn, Salvatore Giorgi, and Niranjan Balasubramanian. 2018. Residualized factor adaptation for community social media prediction tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3560–3569, Brussels, Belgium. Association for Computational Linguistics.

Peter Zeng, Eric Sclafani, and Owen Rambow. 2024. Gram2vec: An interpretable document vectorizer. *arXiv preprint arXiv:2406.12131*.

A Training Details

We experiment with a variety of strategies to decrease training times and GPU memory requirements. All our experiments take place on a server with four 48GB A6000 GPUs. Using the following strategies, our largest model, with approximately 360 million parameters, takes about 5 hours to train. The fastest training time we observed was around 1 hour for our smaller models, which have approximately 150 million parameters. We optimize the model using AdamW (Loshchilov, 2017) with a learning rate of 5e-5, a standard value for fine-tuning pre-trained language models. We train for a maximum of 10 epochs with early stopping based on validation loss to avoid overfitting. With respect to hyperparameters, we manually tune them during the training of RSP. We use these hyperparameters in the rest of our experiments.

We experiment with the use of LoRA (Hu et al., 2021), reducing the number of trainable parameters and lowering memory requirements. Somewhat surprisingly, in our initial experiments fine-tuning RoBERTa for binary classification and for our residual prediction model, performance without LoRA was far lower than performance using LoRA. We hypothesize that LoRA could be acting as a regularizer in this case. We use this to inform our decision of using LoRA in all other experiments in this paper.

Neural Model Contrastive Loss Fine-Tuned Baseline We fine-tune the previously chosen neural models in a Siamese network using a contrastive loss function as our training objective. The architecture for this was heavily inspired by SBERT (Reimers and Gurevych, 2019). We replace SBERT with LUAR or LUAR-RU, and use the pooler output to obtain the embedding for the documents.

Residualized Similarity Prediction Details As RSP is a regression model, we use mean-squared error loss as our training object, and train over 10 epochs. We utilize early stopping to avoid overfitting. We add a regression head with multiple dense layers using ReLU activations and dropout for regularization. We then ensure the output is between -1 and 1 by using a tanh activation.