Residualized Similarity for Faithfully Explainable Authorship Verification

Peter Zeng[♦♠] **Pegah Alipoormolabashi**[♦] **Jihu Mun**[♦] **Gourab Dey**[♦] **Nikita Soni**[♦]
Niranjan Balasubramanian[♦] **Owen Rambow**[♣♠] **H. Andrew Schwartz**[♦]
[♦]Department of Computer Science [♣]Department of Linguistics
[♠]Institute for Advanced Computational Science
Stony Brook University
pezeng@cs.stonybrook.edu

Abstract

Responsible use of *authorship verification (AV)* systems requires not only high-accuracy but also *interpretable* solutions. Specifically, for systems to be deployed in contexts where decisions have real-world consequences, their predictions must be explainable through interpretable features that can be traced to the original text. 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. To address this gap, we introduce *residualized similarity* (RS), [1] a novel method that supplements systems using interpretable features with a neural network to improve their performance while maintaining interpretability. Authorship verification is fundamentally a similarity task, where the goal is to measure how likely two documents are to be written by the same author. The key idea is to use a neural network to predict a *residual similarity*, 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

Identifying the author of a text or a collection of texts is a task with many use cases. In forensic investigations, stylometry techniques and authorship identification help link anonymous social media accounts (Weerasinghe et al., 2022), narrow down suspects (Cafiero and Camps, 2023), and provide supporting evidence in court (Shuy, 1996; M. et al., 2016). Plagiarism and academic dishonesty are cases of intentionally false authorship claims (Enriquez et al., 2023; Kalgutkar et al., 2019).

[1] https://github.com/peterzeng/rsp

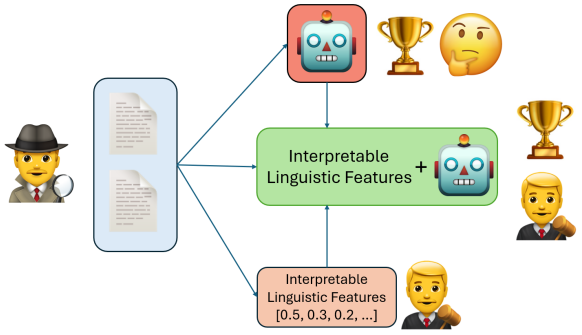

Figure 1: Demonstration of the task of Authorship Verification. A forensic linguist is trying to determine if two texts share the same author. They may use either an interpretable system comprising linguistic features faithful to the source text or a neural model, which has good performance but lacks interpretability. Our system combines the relative strengths of both by using a neural model to correct the error in the interpretable system's prediction.

Forensic and related applications of AI, and of authorship analysis models in particular, require explainable AI (Mersha et al., 2024). All stakeholders need to be able to verify the authorship claims made by any automated system. Furthermore, the explanation must be *faithful*: it "should accurately represent the reasoning process behind the model's prediction" (Lyu et al., 2024). But this is not enough: the reasoning process itself must be based on *interpretable* features derived from the analyzed texts, i.e., the features must be meaningful to humans. Authorship claims from a system using uninterpretable text embeddings cannot be trusted, even if they are used in an explainable reasoning process. But interpretable features are still not enough. Even if the input features are interpretable, they need to be *traceable*. This means that each feature can be traced back to the text(s) being analyzed, and the value of the feature is based on reproducible evidence from the text. A traceable feature cannot be a feature on whose value reasonable observers may disagree after examining the

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 15824–15837
November 4-9, 2025

text, such as “formality".

As with many NLP tasks, authorship systems that use representations derived from neural language models often achieve better verification performance than interpretable representations do (Devlin, 2018; Vaswani, 2017). However, neural representations are limited in many critical domains because they are not directly interpretable. When attempts are made to interpret predictions, such as by 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 traceability of linguistic representations and the high performance of neural models.

As the main contribution of the paper, we introduce *residualized similarity* (RS), which uses the idea of estimating the *residual* of a predictor i.e., the error in a model's prediction. We approach Authorship Verification as a similarity task, as is standard in the field. For each pair of documents we obtain some similarity score from a system; if the score is above a certain threshold, we conclude the documents share the same author, and if the score is below the threshold, we say the documents are from different authors. Suppose we start with an explainable system using interpretable and traceable features 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 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 (Sclafani, 2023) as our interpretable feature system, which records normalized frequencies of morphological and syntactic features for input texts. We evaluate our RS approach by combining Gram2vec with a state-of-the-art AV neural model, LUAR (Rivera-Soto et al., 2021), finding that RS can match the performance of using LUAR alone, while introducing faithful explanations using interpretable and traceable features. 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 7.

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); Huang et al. (2025)).

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. Gram2vec, the interpretable component of residualized similarity falls into this category.

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 of interpretable authorship verification, we focus on these SOTA AV models, particularly LUAR (Rivera-Soto et al., 2021).

Our work uses residual similarity 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 traceable 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

3.1 Problem Statement:

As we argued earlier, we want to develop an authorship verification system for applications such as forensic linguistics, where we want the system to use *traceable interpretable* features as much as possible but also be highly accurate. To this end, we introduce an interpretable variant of the authorship verification problem, which not only requires an output decision but also requires an explicit confidence metric that shows the extent to which the decision can be attributed to the traceable interpretable features. On the one extreme, linguistic feature-based classifiers (e.g., Gram2Vec (Sclafani, 2023)) will have an interpretability confidence that is 100% but lower accuracy, while on the other extreme, a neural embedding-based classifier (Rivera-Soto et al., 2021) will likely have higher accuracy but 0% interpretability confidence. The goal then is to design a system that achieves high accuracy and high interpretability confidence."

3.2 Method Description

To address the above-mentioned problem, we use residualized similarity (RS), whose key idea 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 the standardized pairs of vectorized documents. The ground truth label is 1 for a pair of documents written by the same author and -1 otherwise. RS is trained to predict a residual $r^{(i)} = y^{(i)} - \textbf{sim}(f(d_1), f(d_2))$, 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*.

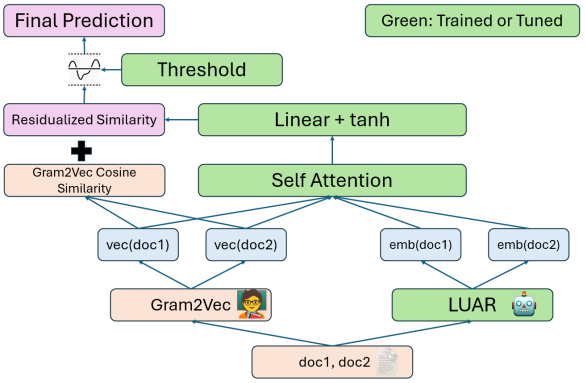

Figure 2: **Residualized Similarity Architecture.** To incorporate signal from the interpretable feature vectors, we add an attention layer over both the interpretable feature vectors as well as the neural embeddings from the model we're fine-tuning. Boxes colored in green indicate that they're updated during training. On the left-hand side, we show the system in use at inference time. The final similarity score is a simple sum of the interpretable cosine similarity score and the predicted residual.

Figure 2 illustrates the specifics of training the RS model and usage at inference time. The process of training RS 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 embeddings, for RS 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 stage. At inference time, given a pair of documents, the final similarity score is a simple summation of the cosine similarity from our interpretable system and the predicted residual.

Training Objective Given a batch of document pairs $\mathcal{B} = \{(d_1^{(i)}, d_2^{(i)}, r^{(i)})\}_{i=1}^{n}$, where $r^{(i)}$ is the ground-truth residual for the pair $(d_1^{(i)}, d_2^{(i)})$, the model aims to predict the residual similarity, denoted as $\hat{r}^{(i)}$. The training objective is to minimize the Mean Squared Error (MSE) between the predicted and actual residuals:

$$\mathcal{L}(\mathcal{B}) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{r}^{(i)} - r^{(i)}\right)^2$$

3.3 Interpretability Confidence

We introduce the notion of *"interpretability confidence"*(IntConf), which is a way to measure how interpretable a particular prediction using residualized similarity is. We define IntConf based on the final prediction. Let the similarity between two texts determined by Gram2vec alone be $\textbf{sim}_{\text{g2v}}$. Thus, IntConf represents the relative contribution of Gram2vec to the overall similarity score, composed of the Gram2vec-derived score and the residual predicted by RS. We distinguish two cases: the RS system predicts that the texts are by the same author, or it predicts different authors. If the prediction is same author, the correct prediction is a high similarity score. The contribution of Gram2vec is the distance from -1, the lowest similarity score possible, so we quantify the contribution of Gram2vec as $1 + \textbf{sim}_{\text{g2v}}$, and divide it by the sum of the Gram2vec contribution and the contribution of the residual component, |predicted residual|. Thus, we get:

$$\frac{1 + \textbf{sim}_{\text{g2v}}}{1 + \textbf{sim}_{\text{g2v}} + |\text{predicted residual}|}$$

(or 0, if $1 + \textbf{sim}_{\text{g2v}} + |\text{predicted residual}| = 0$). If the prediction of the system is different author, then the contribution of the Gram2vec component is the distance from +1, the highest similarity score. Thus, in this case, IntConf is defined as:

$$\frac{1 - \textbf{sim}_{\text{g2v}}}{1 - \textbf{sim}_{\text{g2v}} + |\text{predicted residual}|}$$

The IntConf always takes values between 0 and 1. Note that we can calculate the IntConf for any *specific* pair of documents after running RS.

We note that sometimes the predicted residual "flips" the prediction of the Gram2vec-only system, from "same author" to "different authors" or *v.v.* This possibility is of course precisely why we are predicting the residual. We emphasize that even in cases where the prediction is flipped after using RS we can still make use of the underlying interpretable system for interpretation. We show in section 7 that when the prediction was changed, the underlying interpretable system can help explain why a prediction was made.

4 Experimental Setup

We perform experiments across a variety of neural models, in order to evaluate robustness and ability to generalize of our technique. We specifically leverage sentence-transformer-based models (Reimers and Gurevych, 2019), which were developed as a distinct technology in parallel to generative LLMs. Embedding models specialize in creating fixed-length semantic representations optimized for similarity computations–the way authorship verification is evaluated.

4.1 Models

We evaluate a diverse set of pre-trained encoders to ensure the method is model-agnostic and robust across architectures. We include RoBERTa (Liu, 2019) as a general purpose transformer and Longformer (Beltagy et al., 2020) to handle long documents. We try specialized authorship representation models–LUAR (Rivera-Soto et al., 2021) and a Style-Embedding model, (Wegmann et al., 2022)–both pre-trained to capture distinctive style features. We also test all-mpnet-base-v2 (Song et al., 2020), a sentence-transformer model at the top of the SBERT.net leaderboard, and a more recent model, mxbai-embed-large-v1 (Li and Li, 2023; Lee et al., 2024), which achieves SOTA for BERT-large sized models on the Massive text embedding benchmark (MTEB) (Muennighoff et al., 2023).

4.2 Baselines

We compare residualized similarity against two classes of baselines: the interpretable Gram2vec baseline, and the non-interpretable neural baselines. The models used for the neural baselines are mirrored in the models we tune in our system to predict the residual, all of which have less than one billion parameters. 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, embeddings.

4.3 Residualized Similarity Framework

Let d_1, d_2 be two documents. Let $y \in \{1, -1\}$ be the gold label, where 1 indicates the same author and -1 indicates different authors. Let f represent the Gram2vec vectorizer. Let $\textbf{sim}(v_1, v_2)$ be the cosine similarity function between two vectors v_1 and

v_2. The interpretable system's similarity score is $s = \textbf{sim}(f(d_1), f(d_2))$. The ground truth residual (actual residual) is defined as $\textbf{res\_actual} = y - s$. Let **res_pred** be the residual predicted by a neural model M_{res}. The model M_{res} is trained to approximate **res_actual**. The final similarity score is given by $\textbf{final\_score} = s + \textbf{res\_pred}$. Let t be the classification threshold, which depends on preferences for false positives over false negatives and can be obtained by tuning on a held-out set. The final authorship prediction is obtained by comparing **final_score** to the threshold: same author if greater than t, different author if less than t.

Training Details All neural models and RS 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.

4.4 Data

We train and evaluate our residualized similarity 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 RS and the contrastive-loss fine-tuned baseline, we require the data to be in a labeled paired format: {Document 1, Document 2} and same/different author. 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 curated by ConvoKit (Chang et al., 2020). We use a version preprocessed by (Wegmann et al., 2022), as invalid comments, comments containing only some sort of white space, and deleted comments are removed. We create pairs of comments, label them for author verification. Reddit comments can be naturally very short, so we further filter the comment pairs and keep only comments longer than 20 words. This results in roughly 50,000 training pairs, 10,000 validation pairs, and 10,000 testing pairs. For the rest of the datasets, in order to have comparable train/validation/test sizes, we randomly sample them to match the size of the Reddit train/validation/test splits.

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 respectively. The ratio of same to different author pairs of all datasets is 1:1.

5 Results

Metrics We evaluate RS 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 calculates the true positive rate (TPR) and false positive rate (FPR) at every threshold, and graphs 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.

	Reddit$_{AUC}$		**Amazon**$_{AUC}$		**Fanfiction**$_{AUC}$		**Pikabu**$_{AUC}$	
G2V (Gram2Vec)	0.63		0.71		0.69		0.65	
Neural Models	Neural	RS	Neural	RS	Neural	RS	Neural	RS
RoBERTa-Base	0.69	0.72	0.87	0.85	0.88	0.87	-	-
RoBERTa-Large	0.71	0.70	0.90	0.86	**0.91**	0.86	-	-
Longformer	0.74	0.71	0.88	0.85	0.89	0.86	-	-
LUAR/LUAR-RU	**0.84**	**0.80**	**0.91**	**0.90**	0.89	0.87	**0.74**	**0.76**
Style	0.77	0.76	0.83	0.83	0.72	0.78	-	-
all-mpnet-base-v2	0.66	0.62	0.86	0.84	0.90	0.85	-	-
mxbai-embed-large	0.72	0.69	0.88	0.85	0.87	**0.89**	-	-

Table 1: Model performance across three datasets: Reddit, Amazon, and Fanfiction. Gram2vec represents the fully interpretable baseline, the neural columns represents the non-interpretable contrastive-loss fine-tuned baseline, and Residual is our system. Pikabu is a Russian dataset, and we use LUAR-RU as the neural model in RS (its English counterpart generally performed the strongest in the English datasets). The best performing system in each column is bolded.

5.1 System Evaluation

We show the performance of our RS system in Table 1. The results of Gram2vec alone are given in the first row. We then show the results of testing seven different neural models on three English corpora, and one neural model on a Russian corpus. For each neural model and dataset, we present two results, the first being the neural baseline, and the second being our RS system performance. For all combinations of datasets and models, we see that the uninterpretable neural system and the partially interpretable RS system perform approximately similarly, with no clear pattern emerging with respect to the neural system and/or to the corpus. We see that Gram2vec performs consistently worse than any neural system or any of our RS systems, but above the random baseline (0.5).

We see that LUAR outperforms all other neural models for Reddit and Amazon, and performs competitively for Fanfiction. This is expected because LUAR is trained for authorship attribution. Furthermore, LUAR is particularly good for Reddit, which is also expected, as LUAR is trained on Reddit exclusively. Therefore, we now concentrate on LUAR.

The performance for RS using LUAR and the uninterpretable LUAR neural baseline are nearly identical in AUC for each dataset, with RS performing slightly worse for Reddit, and slightly better Pikabu. However, we observe a big increase in performance compared to using Gram2vec alone, with the biggest improvement being an increase of 19 points on the Amazon dataset.[2]

5.2 Architecture Ablation

We experiment with two alternative residual architectures: (i) a simpler version that passes only the neural embeddings into the regression head, leveraging the representation power of language models like RoBERTa for sequence classification through the [CLS] token, and (ii) a variant that directly appends the interpretable feature vectors to the neural embeddings before passing into the regression head to predict the residual. The latter incorporates signal directly from the interpretable system into the training of RS. We provide an ablation study comparing these two variants with our final version in Table 2.

Data	**RS**	**Only-Neural**	**Appended**
Reddit	0.80	0.73	0.73
Amazon	0.90	0.84	0.85
Fanfiction	0.87	0.74	0.77

Table 2: An ablation study of the different variants of our system. This table is a comparison of our final **residualized similarity (RS)** system, which uses an attention layer to combine the interpretable feature vectors with the neural embeddings, compared to an initial version that passed the neural embeddings directly into the regression head (**Only-Neural**), and another version that concatenated the neural embeddings to the interpretable features directly (**Appended**).

[2]We perform significance tests (paired bootstrap on AUC, two-sided) to compare RS to Gram2Vec, and for all four corpora this difference is significant with $p < 0.001$.

Data	Gram2vec	Gram2vec-RS	ELFEN	ELFEN-RS	Combined	Combined-RS
Reddit	0.63	0.80	0.59	0.80	0.63	0.80
Amazon	0.71	0.90	0.71	0.90	0.74	0.88
Fanfiction	0.69	0.87	0.66	0.88	0.70	0.86

Table 3: The results of a **residualized similarity (RS)** system with Gram2vec, ELFEN, and the concatenation of both as the interpretable system. While the concatenation of the two tends to perform a bit better than either feature set alone, the RS system trained on top of the concatenation performs worse than either RS system trained on a single interpretable system.

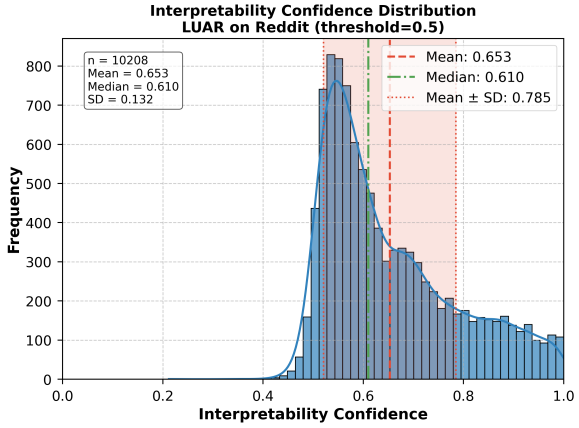

Figure 3: The distribution of interpretability confidence scores in the predictions using residualized similarity with LUAR on the Reddit dataset.

5.3 Analysis of Interpretability Confidence

We plot the distribution of interpretability confidence scores in the predictions using RS with LUAR on the Reddit dataset shown in figure 3. Notably, we observe in most cases, we have an INTCONF of greater than 0.5, and a mean of .65. This suggests that our RS system retains a useful amount of interpretability from the Gram2vec system while increasing performance.

6 Robustness of the Technique

To show that our technique is not dependent on any single interpretable base system, we try another feature extractor in place of Gram2vec. We turn our attention towards Efficient Linguistic Feature Extraction for Natural Language Datasets, or ELFEN (Maurer, 2025). While some features overlap, ELFEN captures several unique categories of features from Gram2vec, and as such, provides a good alternative interpretable system to use in our RS framework. ELFEN feature areas include dependency, emotion, entities, information, lexical richness, morphological, part-of-speech, readability, semantic, and surface. We provide brief explanations of the different features in Appendix C.

We perform experiments using ELFEN in place of Gram2vec as the interpretable system, as well as the concatenation of the two interpretable feature sets as the interpretable system. We perform these experiments using LUAR as the neural model as it is generally the strongest performing neural model of our primary experiments. We find that on its own, ELFEN is a little less performant or on par with Gram2vec, while concatenating the two feature sets tends to improve a bit on either. However, notably, the RS model performs about the same regardless of the interpretable system that it is trained on. We consider this to be a success of the residualized similarity technique as it is able to boost the performance of two separate interpretable systems.[3]

7 Case Study of Two Pairs of Documents

We present two cases to illustrate how RS can give a user insight while performing a specific authorship verification task. We present two pairs of documents, one of which is indeed from the same author, and one of which is not. Given the range of cosine similarity from -1 to 1, we set a threshold of 0.5, representing a moderately strong alignment in vector space and 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

[3]We perform significance tests (paired bootstrap on AUC, two-sided) to compare Gram2vec-RS to Gram2vec alone, ELFEN-RS to ELFEN alone, and Combined-RS to Combined alone. For all three corpora and all three pairs of systems, this difference is significant with $p < 0.001$.

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. Pilae, Obi-Wan, and Anakin 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:
He scanned the field beyond and was dumbfounded when he didn"t see any rats. 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, **RS Predicted Residual**: 0.29, **Final Score**: 0.38
Interpretability Confidence: 0.76 **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, **RS Predicted Residual**: 0.82, **Final Score**: 1.02
Interpretability Confidence: 0.59 **Flipped**: True

Figure 4: Example Pairs for Case Study. Pair 1 is by two different authors, and Pair 2 is by the same author.

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 $n = 5$.

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
morph_tags:Definite=Ind	2.8	2.4	-0.4
func_words:when	2.6	-0.4	2.2
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
punctuation:,	3.4	-1.7	-1.7

Table 4: Top half: The feature scores comparison between the Example 1 document pair by different authors. Bottom half: The feature scores comparison between the Example 2 document pair by the same author.

7.1 Example 1: Different Author Pair

In the first example in Figure 4, both Gram2vec and RS predict that these two documents are written by different authors: the Gram2vec similarity is $0.09 < 0.5$ and RS's is $0.09 + 0.29 = 0.38 < 0.5$, and thus that the label is not flipped. Since the prediction is "different author", we use the formula defined in Section 3.3 to calculate the IntConf as:

$$\frac{1 - 0.09}{1 - 0.09 + 0.29} = 0.76$$

giving us a high confidence in the interpretability. In the top half of Table 4, we show the top 5 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 the top 5 features in descending order. These represent the 5 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). Document 2, in contrast, has few indefinites, and the narration centers on entities known to the readers and the characters in the story.

7.2 Example 2: Same Author Pair

Gram2vec predicts that the two documents are written by different authors, getting the prediction wrong on its own. But overall, RS predicts correctly that these two documents are written by the same author. Since the prediction is "same author", we use the formula defined in Section 3.3 to calculate the INTCONF as:

$$\frac{1 + 0.20}{1 + 0.20 + 0.82} = 0.59$$

giving us a moderately high confidence in the interpretability. Even though the label was flipped from Gram2vec to RS in this case, we observe that there are still 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 5 features in descending order, shown in the bottom half of Table 4. 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). The two documents also use passive voice clauses more frequently than on average: *it was connected* (document 1), *it was passed down*, *it was created* (document 2). Both of these linguistic features are relatively rare in standard written English. 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.

We note that this paper does not propose an end-to-end explainable system. Instead, we have shown how our RS 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.

8 Conclusion

We introduce residualized similarity, 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. We apply this technique to the task of authorship verification, where interpretability is of the utmost importance. Using residualized similarity, we are able to achieve state-of-the-art performance on the task of authorship verification while maintaining a quantifiable 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 RS, 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 RS agreed, we show that there is meaningful interpretability in the features, as well as the ability to trace such features back to the text from which they were extracted.

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 (RS), 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.

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 evaluating 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.

Ethics Statement

The underlying 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.

Acknowledgements

This material is based upon work supported in part by the National Science Foundation (NSF) under No. 2125295 (NRT-HDR: Detecting and Addressing Bias in Data, Humans, and Institutions); as well as by the Intelligence Advanced Research Projects Activity (IARPA) under the HIATUS program (contract 2022-22072200005). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the NSF, DARPA, or IARPA.

We thank both the Institute for Advanced Computational Science and the Institute for AI-Driven Discovery and Innovation at Stony Brook for access to the computing resources needed for this work. These resources were made possible by NSF grant No. 1531492 (SeaWulf HPC cluster maintained by Research Computing and Cyberinfrastructure) and NSF grant No. 1919752 (Major Research Infrastructure program), respectively.

We thank our ARR reviewers, whose comments have contributed to improving the paper.

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*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

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.

Florian Cafiero and Jean-Baptiste Camps. 2023. Who could be behind qanon? authorship attribution with supervised machine-learning. *Digital Scholarship in the Humanities*, 38(4):1418–1430.

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).

Daniel Enriquez, Gage Christensen, Hayden Donovan, Jared Lam, Noah Wong, Sergiu Dascalu, David Feil-Seifer, and Emily Hand. 2023. Authorship verification for hired plagiarism detection. In *Proceedings of the 9th International Conference on Applied Computing & Information Technology*, ACIT '22, page 19–24, New York, NY, USA. Association for Computing Machinery.

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*.

Baixiang Huang, Canyu Chen, and Kai Shu. 2025. Authorship attribution in the era of llms: Problems, methodologies, and challenges. *SIGKDD Explor. Newsl.*, 26(2):21–43.

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).

Vaibhavi Kalgutkar, Ratinder Kaur, Hugo Gonzalez, Natalia Stakhanova, and Alina Matyukhina. 2019. Code authorship attribution: Methods and challenges. *ACM Comput. Surv.*, 52(1).

Sean Lee, Aamir Shakir, Darius Koenig, and Julius Lipp. 2024. Open source strikes bread - new fluffy embeddings model. Huggingface.

Xianming Li and Jing Li. 2023. Angle-optimized text embeddings. *arXiv preprint arXiv:2309.12871*.

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.

Coulthard M., Johnson A., and Wright D. 2016. *An Introduction to Forensic Linguistics: Language in Evidence (2nd ed.)*. Routledge.

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*.

Maximilian Maurer. 2025. Elfen - efficient linguistic feature extraction for natural language datasets. `https://github.com/mmmaurer/elfen`.

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

Melkamu Mersha, Khang Lam, Joseph Wood, Ali AlShami, and Jugal Kalita. 2024. Explainable artificial intelligence: A survey of needs, techniques, applications, and future direction. *Neurocomputing*, page 128111.

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

Saif Mohammad. 2018a. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 English words. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 174–184, Melbourne, Australia. Association for Computational Linguistics.

Saif Mohammad. 2018b. Word affect intensities. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Saif M Mohammad and Peter D Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational intelligence*, 29(3):436–465.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

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.

Eric Sclafani. 2023. Gram2vec.

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.

Roger W. Shuy. 1996. *Language Crimes: The Use and Abuse of Language Evidence in the Courtroom*. Wiley-Blackwell, Oxford, UK.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. Mpnet: Masked and permuted pre-training for language understanding. *Advances in neural information processing systems*, 33:16857–16867.

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*.

Janith Weerasinghe, Rhia Singh, and Rachel Greenstadt. 2022. Using authorship verification to mitigate abuse in online communities. *Proceedings of the International AAAI Conference on Web and Social Media*, 16(1):1075–1086.

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.

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 RS. 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 to use 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 Details As RS is a regression model, we use mean-squared error loss as our training objective, and train over 10 epochs. We utilize early stopping to avoid over-fitting. 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.

B Cross-Domain Experiments

We evaluate RS models trained on one domain on the other domains, similar to the cross-domain experiments in the LUAR paper (Rivera-Soto et al., 2021). In addition, we evaluate a RS model trained on the English Reddit data on the Russian Pikabu data to observe if there is any cross-lingual transfer.

Trained On	Dataset	G2V	RSP
Reddit	Reddit	0.63	0.80
	Amazon	0.71	0.81
	Fanfiction	0.69	0.71
Amazon	Reddit	0.63	0.71
	Amazon	0.71	0.90
	Fanfiction	0.69	0.72
Fanfiction	Reddit	0.63	0.67
	Amazon	0.71	0.75
	Fanfiction	0.69	0.87
Reddit	Pikabu	0.65	0.62

Table 5: Cross-domain experiments. Unsurprisingly, models trained on one domain tend to do a bit worse when evaluating on the other two domains. However, the performance in all cases still beat using just Gram2vec. We also witness no cross-lingual transfer, suggesting a model trained on English data would not be able to help in evaluation performance on Russian data.

C ELFEN Feature Categories

Efficient Linguistic Feature Extraction for Natural Language Datasets (ELFEN) (Maurer, 2025) is an interpretable feature extractor. We provide brief explanations of the different feature types; the full feature list can be found at the github[4].

- Dependency: dependency features are based on the dependency tree of the text data, with features such as tree width, tree depth, and more.
- Emotion: emotion features include valence, arousal, dominance dimensions, Plutchik emotions, the sentiment and emotion intensity of text data. Calculating these features uses the NRC VAD Lexicon (Mohammad and Turney, 2013), the NRC Emotion Intensity Lexicon (Mohammad, 2018a), and the NRC Sentiment Lexicon (Mohammad, 2018b).
- Entities: entity features include named entity-related features such as 'PERSON', 'MONEY', 'PRODUCT', 'TIME', and 'PERCENT'.
- Information: this area includes information-theoretic metrics, in particular, compressibility and Shannon Entropy.

[4]https://github.com/mmmaurer/elfen/blob/main/features.md

- Lexical Richness: this area contains various lexical richness metrics such as lemma/token ratio, type/token ratio, and root type/token ratio.
- Morphological: this area captures the counts of tokens with a specific type of morphological tag such as "VerbForm" or "Number".
- Part-of-Speech: this area includes various part-of-speech (POS) related features such as POS variability and number of tokens per POS tag.
- Readability: this area includes different readability/complexity scores such as number of syllables and number of mono/poly-syllables.
- Semantic: this area includes different types of semantic features such as number of hedge words and ratio of hedge words.
- Surface: this area includes “surface-level" features such as raw sequence length, number of tokens, and number of sentences.