Residualized Similarity Prediction for Maintaining Interpretability in Authorship Verification

Anonymous ACL submission

Abstract

Responsible use of authorship verification systems not only requires high accuracies but also *interpretable* solutions. Neural methods achieve high accuracies but their representations lack direct interpretability, whereas methods using interpretable linguistic features generally perform worse than neural methods. 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 *residual similarity*, i.e. the error in the similarity predicted by the interpretable system. Our evaluation on three datasets shows that using RSP improves authorship verification predictions over a fully interpretable system, multiple neural models, as well as weighted ensembles of these two (RSP yields gains in 17 of the 24 combinations), all while maintaining interpretability as measured using a new interpretability confidence metric.

1 Introduction

Authorship verification is a task with many critical applications such as plagiarism detection, forensic linguistics, and literary analysis. Responsible and ethical development of these applications demands, among others, *interpretable* solutions, ones where the representations used for verification are simple aggregates of relevant indicators that are used by practitioners and readily understood by stakeholders. For example, forensic linguists may rely on linguistic indicators to justify authorship verification. As with many NLP tasks, representations derived from neural language models often achieve better verification performance than interpretable representations do. However, these neural representations are not directly interpretable, seriously limiting applicability in many critical domains.

In this paper, we ask how one can combine the relative strengths of the two methods: the interpretability of representations and the high performance of neural models. One way of doing so is direct ensembling, which involves determining fixed weights that combine scores from both an interpretable system (i.e., a system which uses only interpretable representations) and a neural system. However, when this ensembling method is optimized for performance, the weight of the interpretable system is small, and when the the method is optimized for interpretability, the performance decreases. What we want, instead, is a more dynamic approach, one where we can rely on the interpretable system more when it is likely to be accurate, and rely on the neural model otherwise.

To realize this, 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 first train an interpretable system as the main similarity predictor. We can then train a neural model as a *residual predictor*, which predicts the error or correction to the interpretable model's predicted similarity. The final prediction is a simple sum of the prediction from the interpretable model and the residual, i.e., the correction, predicted by the neural model. This combined system achieves the trade-off we desire: (i) when the interpretable model is likely to be correct, the residual should be near zero, providing full interpretability and 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 prior work by Zamani et al. (2018), who trained residual models for a regression problem, combining linguistic and health-relevant attributes for predicting community health indicators.

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 RSP approach by combining Gram2vec with various neural models trained to predict the *residuals*. We show that RSP improves under most conditions, and establishes a new SOTA on one of three genres. Our system retains interpretability, measured by an *interpretability confidence* metric, which indicates the extent to which the interpretable system is used for a given input.

2 Related Work

Authorship verification, authorship attribution, and authorship profiling are part of authorship analysis which has been explored through a wide range of approaches (see surveys El and Kassou (2014); Misini et al. (2022)). Here we discuss interpretable methods that make use of stylometric features and recent neural models. **(i) 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. (ii) **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 Longformer (Ordoñez et al., 2020; Nguyen et al., 2023). More recently, sentence-transformer based models (Wegmann et al., 2022; Rivera-Soto et al., 2021) obtain state-of-the-art performance for AV tasks.

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 of health outcome prediction 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). However, in this work, our focus is only on combining interpretable and neural models and not on generating explanations. 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). We can also treat these as interpretable systems (if they are faithful) and combine with other neural models to further improve performance.

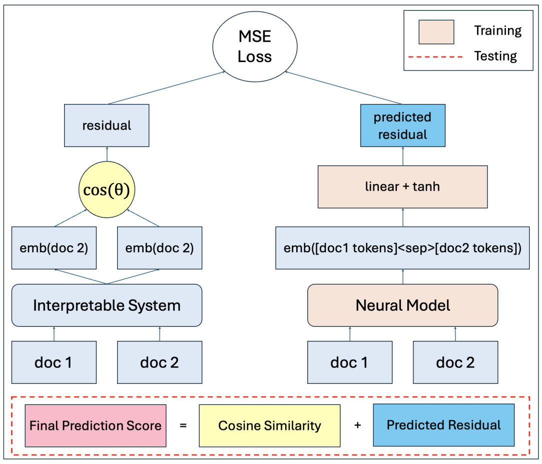

Figure 1: **Residualized Similarity Architecture.** Note, the left side of the diagram is not updated during training, and merely provides labels for the model to learn. We add a sequential layer, alternating linear and ReLU layers onto the encoder model to output the regression value, which we then pass through a tanh activation function. This is done to introduce non-linearity and capture more rich information in our fine-tuning.

3 Residualized Similarity Prediction

The key idea in residualized similarity prediction is to train a neural model to predict the residual, i.e., the difference between the cosine similarity obtained from our interpretable system and the ground truth. We generate interpretable feature vectors for each document using Gram2vec and calculate their cosine similarity. Since these feature vectors store normalized counts of grammatical features, the cosine value is always non-negative. The ground truth label is 1 for a pair of documents written by the same author and 0 otherwise. We train the neural residual model to predict $y - \text{sim}(f(d_1), f(d_2))$ where y is the gold label, d_1 and d_2 are the two documents, and f is the Gram2vec vector function.

Figure 1 illustrates the specifics of training the RSP model. Note that the left half of the figure does not involve any trainable parameters. During training, this part produces the residuals needed for training the right side of the figure. The trained part includes the neural model and a simple linear layer with a $\tanh$ non-linearity to produce residuals in the range $(-1, 1)$. We train a variety of neural models for our experiments, and plan on releasing our code publicly.

4 Experimental Setup

Our evaluation is aimed at testing how the residualized similarity prediction method fares against the two methods it combines: an interpretable system, neural models fine-tuned on the target datasets, as well as a weighted ensemble of the two.

4.1 Methods

Gram2vec System: We use Gram2vec to derive interpretable feature vectors from texts. These vectors comprise normalized relative frequencies of various grammatical features of documents, such as part-of-speech tag unigrams and bigrams, morphology tags, dependency labels, and more. We then compute cosine similarity between the two vectors. If the cosine similarity exceeds a specific threshold (tuned on the training data), we label the input pair as being from the "same author"; otherwise, we label them as being "from different authors".

Neural Models: We train four neural models: RoBERTa (Liu et al., 2019), both base and large versions, Longformer (Beltagy et al., 2020) which has been designed for long contexts such as the document pairs needed in AV, and the SOTA LUAR (Rivera-Soto et al., 2021) model, an SBERT (Reimers and Gurevych, 2019) embedding model trained specifically for authorship tasks. We use these neural models in two modes: **(i) Classification System**, where we train them as binary classifiers to predict same or different author labels. This setup is aimed to show the best performance one can achieve with the neural model alone when it is trained on the target set. **(ii) Cosine System**, where we train them to produce document embedding (vectors), whose cosine similarity is thresholded to produce same or different author labels.

Ensemble: We use a weighted average of the cosine similarities from the Gram2vec and neural systems. The tuned parameter λ indicates the contribution of Gram2vec.

Residualized Similarity Prediction: We train each neural model on residuals obtained from the training set using Gram2vec similarities. During inference, the sum of Gram2vec's cosine similarity and the predicted residual is thresholded for producing the class labels.

Training Details: All neural models and RSP are trained using LoRA (Hu et al., 2021), which not only reduces the number of trainable parameters and memory requirements, but also yields better performance overall for all models. Thresholds are selected from (-1,1) and the ensemble's λ is selected from (0,1) both in increments of 0.1. All tuning for the threshold and λ are performed (separately for each system) on the training set. Additional training details can be found in Appendix C.

4.2 Data

We train and evaluate our model on three datasets covering diverse genres: **(i) Reddit comments:** We use a version preprocessed by (Wegmann et al., 2022) with invalid comments removed from the original Reddit comments from 100 active subreddits created by ConvoKit (Chang et al., 2020). We filter pairs that contain short comments (less than 20 words). **(ii) Amazon reviews:** We create document pairs from three categories in the original dataset (Ni et al., 2019): Office Products; Patio, Lawn and Garden; and Video games. We only use authors who have at least two reviews of twenty or more words. **(iii) Fanfiction Stories:** We use a paragraph version of the original stories dataset (Bischoff et al., 2020). Since the stories can be long, we split them into paragraphs following the setup described in Rivera-Soto et al. (2021).

For all three datasets, we use 50K, 10K, and 10K pairs for the training, validation, and test sets respectively. The ratio of same to different author pairs is 1:1. Appendix B has additional details.

5 Results

We evaluate RSP against the neural classification system on same-author F1 score, as we consider same author verification the primary goal of these models. We evaluate RSP against the neural cosine systems (all systems that use a threshold) on AUC. Our results are detailed in Table 1.

(i) RSP improves F1 or AUC in most (dataset, neural model, system type) conditions: RSP improves same author F1 and AUC greatly compared to using the interpretable Gram2vec system alone. Furthermore, when compared to the neural models, RSP generally improves over the non-LUAR neural models: of the 24 individual results, RSP performs best in 17 (shown in bold in Table 1).

(ii) RSP is better than ensembling: In 8 of 12 cases, the weighted averaged ensembling, a standard way to combine two models, fares worse than RSP on AUC despite exhaustive grid search of both λ and the similarity threshold. The low λ values further show that ensembling heavily favors the

		Classification System, F1			Cosine Systems, AUC			
Dataset	Neural Model	G2V	Neural	RSP	G2V	Neural	Ensemble (λ)	RSP
Reddit	RoBERTa-base	0.67	0.66	**0.69**	0.57	0.69	0.69 (0.13)	**0.73**
	RoBERTa-large		0.69	**0.71**		0.73	0.73 (0.10)	**0.77**
	Longformer		0.68	**0.70**		0.71	0.72 (0.14)	**0.75**
	LUAR		**0.74**	0.70		0.80	**0.84** (0.04)	0.73
Amazon	RoBERTa-base	0.67	0.77	**0.81**	0.61	0.86	0.86 (0.08)	**0.88**
	RoBERTa-large		**0.84**	0.83		0.89	0.89 (0.06)	**0.90**
	Longformer		0.78	**0.81**		**0.87**	0.87 (0.09)	0.86
	LUAR		**0.78**	**0.78**		**0.92**	0.92 (0.0)	0.84
Fanfiction	RoBERTa-base	0.67	0.73	**0.81**	0.56	0.83	0.84 (0.11)	**0.87**
	RoBERTa-large		0.83	**0.86**		0.88	0.88 (0.06)	**0.91**
	Longformer		0.74	**0.83**		0.85	0.85 (0.08)	**0.89**
	LUAR		**0.74**	0.70		**0.88**	0.88 (0.0)	0.74

Table 1: Comparison of a neural finetuned classifier against our **residualized similarity prediction** (RSP) system using same author F1, and a neural cosine embedding and ensemble of that with Gram2vec, also against RSP, using same author AUC. G2V = Gram2vec. The best performing system for each combination of dataset, neural model, and system type (classification or cosine) is **bold**; the best performing system for each combination of dataset and system type, i.e., across neural models, is shaded. If $\lambda = 0$, the ensemble system is the same as the neural system. Residualized similarity shows the highest consistency for top results for a majority of neural models as well across domains, while best performing models overall were split between fully neural, ensemble, and RSP approaches.

uninterpretable neural model; the RSP model can *softly* retain interpretability as much as possible.

(iii) Comparison to SOTA: LUAR currently represents the state-of-the-art in authorship verification. When we only consider (dataset, system type) conditions, RSP creates new SOTA results in two of the six cases (shown shaded in Table 1), both Fanfiction. RSP system is close to LUAR's AUC in the other datasets (-0.03 in Reddit, and -0.02 in Amazon), while also maintaining interpretability.

5.1 Interpretability Analysis

Even when RSP performs worse than a neural system (usually LUAR), the performance drop is small and RSP retains a measure of interpretability. In order to quantify how much interpretability a specific result retains, we introduce a notion of "interpretability confidence" (IntConf), which is a way to measure how interpretable a particular prediction from RSP is. We define IntConf to have 2 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). We note that we can calculate the IntConf for a specific pair of documents after running the residual system. We further analyze the distribution of the IntConf values and the predicted residuals when using RobERTa-base on the Reddit dataset. We find the mean of the IntConf to be 0.83, showing that on average, the final prediction remains highly interpretable. The mean of the predicted residuals is -0.06, while the standard deviation is 0.195. This shows that RSP is indeed learning when to correct the initial prediction, and applies non-trivial amounts of correction some times. See Appendix D for these distributions and an example calculation of IntConf.

6 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 cosine similarity from an interpretable system and the ground truth. Our experiments on authorship verification across 3 datasets improve results compared to the interpretable system alone, and overall perform well against neural systems and ensembling methods, while maintaining interpretability.

To measure interpretability, we introduce the **interpretability confidence**, a measure of how interpretable a prediction from our system is. 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.

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

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.

Sebastian Bischoff, Niklas Deckers, Marcel Schliebs, Ben Thies, Matthias Hagen, Efstathios Stamatatos, Benno Stein, and Martin Potthast. 2020. The Importance of Suppressing Domain Style in Authorship Analysis. *CoRR*, abs/2005.14714.

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.

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.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

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.

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.

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.

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 Model Details

We train several transformer models for regression to predict the residual between the true label and the cosine similarity from Gram2Vec vectors, binary classification of AV, and to produce embeddings to calculate cosine similarity with. We perform fine-tuning on RoBERTa-base and RoBERTa-large, Longformer, and LUAR — The first two are strong sequence classification models, Longformer (Beltagy et al., 2020) is a RoBERTa-based model that utilizes a sliding window of attention, allowing for much longer contexts (we choose a maximum context length that is twice that of the other selected models), and finally, LUAR is a state-of-the-art attention based authorship verification model built from SBERT.

In this paper, there are two types of AV prediction systems. The **first** type predicts the same author label if the cosine similarity between the embeddings of the input documents exceeds a (fixed) threshold. The threshold is chosen based on training data. Our residual system falls into this category. The **second** type of system includes models fine-tuned for binary classification, labeling document pairs as written by the same or different authors. To get a robust baseline of methods to compare our system to, we decide to obtain a wide range of baselines as follows and mark them with their respective system type(s):

- Cosine similarity between feature vectors from Gram2vec alone (1)
- Cosine similarity between embeddings from the neural models alone, fine-tuned to produce embeddings for authorship verification (1)
- Ensemble method of the first two methods by weighting them and adding the scores. (1)
- Fine-tuning the neural models to perform binary classification (2)

B Dataset Details

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 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. There are comments from about 36K authors in train set, and 7K authors in development and test sets each.

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 preprocessing 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.

C 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. 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.

While we choose Longformer for its ability to capture patterns in longer documents, we found that fine-tuning Longformer takes far longer than the other models. To mitigate this, we set the maximum context length of Longformer to 1024, twice as long as the maximum context lengths of the other models.

Neural Model Binary Classification Baseline To get a sense of how neural models perform when fine-tuned directly for the task of AV, we fine-tune them for binary classification. We add a classification head with 2 classes and use cross entropy loss as our training objective. This model shares training strategies that RSP used including LoRA and early stopping.

Neural Model Cosine 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). Of course, we replace BERT with various different neural models, 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.

D Residual Prediction and IntConf Distributions

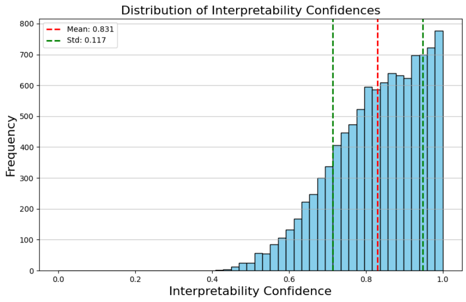

Figure 2: Distribution of interpretability confidences for RSP using RoBERTa on the Reddit dataset.

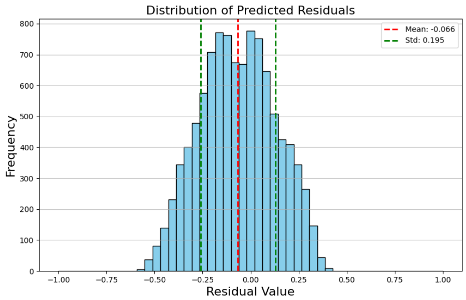

Figure 3: Distribution of predicted residuals for RSP using RoBERTa on the Reddit dataset.

G2V	**Resid.**	**Corr.**	**IC (F)**
0.730	0.062	0.792	0.938 (0)
0.437	-0.265	0.172	0.735 (0)
0.650	-0.216	0.434	0.794 (1)

Table 2: Examples of interpretability confidence calculation for a threshold of 0.5. G2V = Gram2vec; Resid. = predicted residual; Corr = corrected prediction, i.e., G2V + Resid.; IC = Interpretability Coefficient; F = Flipped Indicator