# Neural Normalized Compression Distance and the Disconnect Between Compression and Classification

**John Hurwitz**
University of Maryland, Baltimore County
Laboratory for Advanced Cybersecurity Research

**Charles Nicholas**
University of Maryland, Baltimore County

**Edward Raff**
Booz Allen Hamilton
University of Maryland, Baltimore County

## Abstract

It is generally well understood that predictive classification and compression are intrinsically related concepts in information theory. Indeed, many deep learning methods are explained as learning a kind of compression, and that better compression leads to better performance. We interrogate this hypothesis via the Normalized Compression Distance (NCD), which explicitly relies on compression as the means of measuring similarity between sequences and thus enables nearest-neighbor classification. By turning popular large language models (LLMs) into lossless compressors, we develop a *Neural* NCD and compare LLMs to classic general-purpose algorithms like gzip. In doing so, we find that classification accuracy is not predictable by compression rate alone, among other empirical aberrations not predicted by current understanding. Our results imply that our intuition on what it means for a neural network to "compress" and what is needed for effective classification are not yet well understood.

## 1 Introduction

The link between compression and prediction has been well-established [1]. Predictors (such as machine learning models) can be turned into compressors when combined with an entropy coding technique such as arithmetic coding [2]. Conversely, compressors can be used to make predictions via a compressor-based distance metric and a distance-based learning algorithm such as $k$-nearest neighbors. A popular method for the latter approach is the Normalized Compression Distance (NCD) [3], which is theoretically optimal as a distance metric if given an optimal compressor. NCD has seen success in a variety of domains, including anomaly detection and clustering [4], text classification [5], and image classification [6]. Here, the compressors used are primarily traditional compression algorithms such as *gzip* and *lzma*. While these traditional compression algorithms can compress data quickly and successfully in an impressively wide variety of domains, much better lossless compression performance is generally possible in any given specialized domain through learned neural network-based compressors [7, 8, 9]. In the text domain, the leading compressors in terms of compression rate are neural-network based [10] and far surpass the performance of traditional compressors, at the cost of being more computationally expensive.

Given the superior compression ability of neural compressors, the natural question arises of whether their improved compression rates translate into improved predictive performance. Indeed, it has been hypothesized that compression algorithms with better compression rates should classify better on account of closer approximation to Kolmogorov complexity. We investigate NCD-based classification

Accepted to Machine Learning and Compression Workshop at 38th Conference on Neural Information Processing Systems (NeurIPS 2024).

performance differences by replacing traditional compressors with LLM-based neural compressors in this paradigm, noting the effect of compression rate on test accuracy, and comparing NCD performance with Euclidean distance with a model's latent representation. We find that counter to conventional wisdom, NCD-based classification accuracy is not predictable solely from compression rate. We exhibit cases where a neural compressor achieves consistent and superior compression rates across datasets but can either outperform or underperform traditional compressors depending on the dataset.

This paper is organized as follows. In section 2 we describe our novel approach to doing NCD-based text classification via pretrained LLMs as neural compressors in the few-shot text classification setting, and provide requisite background in the information-theoretic grounding of compression distance. In section 3 we highlight related work in neural compression, compression-distance-based machine learning, and the connection between compression rate and classification accuracy. In section 4 we describe our experimental results and three primary findings relating to the relationship between compression rate and accuracy, effects of varying the choice of LLM for neural compression, and a comparison to using the models' latent representations with Euclidean distance. We then provide a brief conclusion in section 5.

## 2 Method: Neural Normalized Compression Distance

In this section we provide a brief description of Kolmogorov complexity and its connection to practical notions of compression distance, arithmetic coding and its usefulness in lossless compression schemes, and our novel approach to using pretrained LLMs as neural compressors in the few shot text classification setting.

### 2.1 Kolmogorov Complexity and Compression Distance

Given a sequence $x$, the Kolmogorov Complexity $K(x)$ [11] is defined as the length of the shortest computer program that outputs $x$. $K(x)$ can be interpreted as the output of the best possible lossless compressor given any input bitstring $x$. The Conditional Kolmogorov Complexity function $K(x|y)$ is defined as the length of the shortest program that outputs $x$ given another sequence $y$ as input. Using the notion of Kolmogorov Complexity, Li et al. [3] define the Normalized Information Distance (NID).

$$\text{NID}(x, y) = \frac{\max\{K(x|y), K(y|x)\}}{\max\{K(x), K(y)\}} \tag{1}$$

The NID is a metric[1] and has a range of $[0, 1]$. The lower the NID, the more shared information exists between $x$ and $y$ which gives an indication of similarity. It is well-known that $K$ is uncomputable. In practice, we can approximate $K$ by using any compression algorithm. Li et al. [3] define the Normalized Compression Distance such that, given a compression algorithm, and a function $C(x)$ which returns the length of the compressed output of sequence $x$ in bytes, gives us a practical approximation of the NID. $xy$ indicates the concatenation of sequences $x$ and $y$.

$$\text{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}} \tag{2}$$

### 2.2 NCD-Based Sequence Classification

By using any compressor to calculate pairwise NCDs, these NCDs can be used as the distance metric in the non-parametric classification algorithm $k$ nearest neighbors ($k$NN). There is an $O(n^2)$ complexity for the pairwise distance calculations between the train set and test set, each requiring the compression of a concatenated pair of samples. In current practice, this complexity is a limiting factor for choice of model size and dataset size. To use a compressor in this way only requires the compression phase; decompression is never needed.

[1]The proof that NID is a metric includes negligible error terms for the identity axiom and triangle inequality [3].

### 2.3 Arithmetic Coding

In modern neural lossless compression paradigms, arithmetic coding [2] is one of the most popular schemes for entropy coding due to its near-optimality, and its ability to handle adaptive probabilistic models [12, 13, 14, 8, 7]. Arithmetic coding is an entropy coding technique which, given a probability distribution as input, represents a sequence of symbols as a single arbitrary-precision floating-point number between 0 and 1. Symbols with high probability will be encoded with fewer bits than symbols with low probability. Arithmetic coding is compatible with an adaptive probabilistic model where the next-symbol probabilities can differ for each index in the sequence. Language models fit precisely this paradigm, where the prediction at each index of a sequence yields a probability distribution for the next symbol. By doing arithmetic coding on this probability distribution as input, we perform lossless compression. The resulting bitstream, with access to the same probabilistic model, can be used in a reverse process to reconstruct the original sequence.

### 2.4 Neural Normalized Compression Distance for Sequence Classification

We refer to the use of neural network-based compressors to calculate NCDs as Neural Normalized Compression Distance (Neural NCD). We extend previous work on compression-based sequence classification by using Neural NCD in the few-shot text classification setting. The neural compressor, which is a language model paired with arithmetic coding, is used to calculate NCDs that are then used as the distance metric for $k$NN. There are two primary reasons why using pretrained LLMs as the probabilistic models in a lossless compression scheme is useful for studying the relationship between compression rate and accuracy of NCD-based classification. First, neural compressors can achieve much better compression rates than traditional compressors in specific domains, allowing us to investigate the effects of compression rate on classification accuracy when using neural compressors. Second, the model can be easily swapped out, allowing us to study similar effects when varying model architecture and size.

## 3 Related Work

**Neural Compression:** Neural networks have been successfully used to perform lossless compression on a variety of domains including text [14, 7, 8] and images [6]. Large Language Models (LLMs) in particular have shown promise for lossless compression of text [12, 13] due to excellence in next-symbol prediction, with neural methods leading the Large Text Classification Benchmark [10] and achieving better compression rates than traditional compressors like *gzip*, at the cost of more computation. Other works that attempt alternative architectures inspired by compression algorithms are beyond the current scope of this article [15].

**Compression Distance for Machine Learning:** There is empirical evidence to show that compression distance is effective in anomaly detection, clustering, and classification of sequential data [4, 16]. Li et al. [3], who proposed NID, also showed its effectiveness in text classification. Jiang et al. [5] show that NCD-based classification with the traditional compressor *gzip* performs competitively with neural methods such as BERT [17] on few-shot text classification tasks. A follow-up work [6] uses deep latent variable models for compression of images, outperforming supervised neural network methods for few-shot image classification. Others have relaxed the compression requirements and instead extracted feature-vectors from compression dictionaries/methods to obtain faster distance calculations [18, 19].

**Compression Rate and Classification Accuracy:** Jiang et al. [5] show a moderate linear correlation between compression rate and test accuracy among traditional compressors for few shot text classification, while noting the importance of the actual compression algorithm and finding cases where algorithms with better compression rates still under-perform ones with lower ratios. A follow-up work [6] also shows a similar correlation among mostly traditional compressors and one latent variable neural compressor on image classification datasets, and older work in malware analysis found a similar result[20]. Notably, compression similarity has seen wide use in malware for its ability to handle hard-to-parse and large binary files [21, 22, 19, 23, 18, 24].

# 4 Experiments

Three key insights are derived from our study. First, accuracy of NCD-based classification methods is not predictable from compression rate alone when using neural compressors in the text classification domain. Second, by comparing different LLMs as neural compressors, we show that models of similar size but different architectures and pretraining have similar performance when compared to traditional compressors. Third, the extent to which Neural NCD improves or hinders performance over a Euclidean distance approach using the model's latent representations is model-dependent; we show that Neural NCD performs better than latent representations for certain models and performs worse for others.

We investigate Neural NCD for non-parametric NCD-based text classification in a few-shot setting using the datasets AGNews, 20News, and DBpedia (though we only use AGNews and DBpedia in subsection 4.2 and subsection 4.3). We use Bellard's *ts_zip* utility [14] to perform neural compression with the model RWKV 169M [25]. We compare these results with the use of the traditional compressors *gzip*, *zstandard* (*zstd*), and *lzma* on the same task.

To investigate the few-shot setting, we draw $n = \{5, 10, 50, 100\}$ labeled train samples from each class, obtain a subset of 100 test samples via stratified sampling across five trials for each $n$, calculating the mean test accuracy, 95% confidence interval, and compression rate (compressed size / original size). The $O(n^2)$ computational complexity of pairwise NCD calculation for neural compression is a limiting factor in the computational cost of this method, scaling with LLM model size and size of training and test sets. For this reason we subsample only 100 test samples here for each of the five trials. See Appendix A for further experiment details.

### 4.1 Neural compressor has variable Neural NCD performance despite consistently superior compression rates.

It has been hypothesized that compressors that achieve better compression rates should improve the performance of NCD-based methods through a better approximation of Kolmogorov Complexity [5, 6]. Through our text classification experiments with a neural compressor, we find that compression rate alone is not enough to predict the differences in NCD-based classification accuracy when using neural compressors across various datasets in the text domain. We note the compression rates (lower is better compression) for each compressor on each dataset in Table 1.

Ignoring the size of the language models, even a small, simple model such as RWKV 169M achieves significantly lower compression rates than traditional compressors. Despite the RWKV neural compressor achieving superior compression rates than traditional compressors on each dataset, Neural NCD yields varying relative performance when compared to using NCDs of traditional compressors, outperforming them on AGNews, performing on-par with them on 20News, and underperforming them on DBpedia (see Figure 1). Of particular note is that the RWKV compressor and *gzip* retain their compression rates across both AGNews and DBpedia, and yet neural compression outperforms *gzip* on the former and underperforms it on the latter.

Table 1: Compression rates (lower is better compression) across compressors for the AGNews, 20News, and DBpedia datasets. For neural compressors, we use the raw compression rate which ignores the size of the neural network. The RWKV 169M model achieves the best compression rate across each dataset.

| Compressor | AGNews | 20News | DBpedia |
|---|---|---|---|
| gzip | 0.785 | 0.593 | 0.824 |
| zstd | 0.743 | 0.596 | 0.785 |
| lzma | 1.070 | 0.685 | 1.115 |
| RWKV 169M | **0.248** | **0.204** | **0.248** |

We see the same phenomenon strictly among the traditional compressors as well, where there is no big difference in performance despite varying compression rates [2]. We plot test accuracies across compression rates for AGNews and DBpedia across all few-shot settings in Figure 2. We can see that each compressor tends to have its own range of compression rates for these datasets, and despite

[2]Note that for short sequences, LZMA's dictionary size outweighs the compression savings, leading to rates greater than 1. However, despite these poor compression rates LZMA is still able to do NCD-based sequence classification on-par with *gzip* and *zstd*.

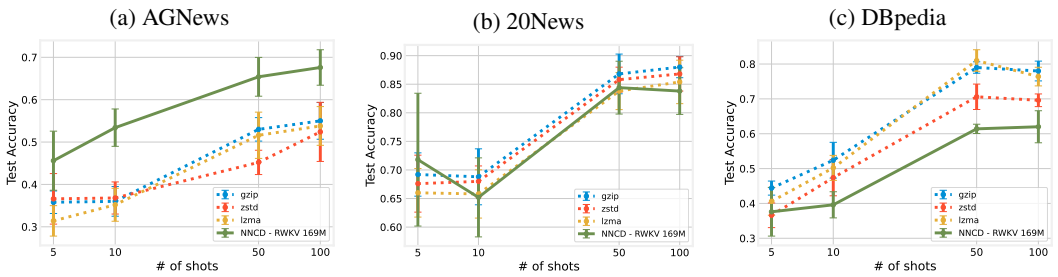


Figure 1: Comparison of RWKV 169M neural compressor and traditional compressors using $k$NN with NCD across the datasets AGNews, 20News, and DBpedia (NNCD = Neural NCD). Despite the neural compressor achieving superior compression rates, we find cases where Neural NCD outperforms, underperforms, and performs on-par with traditional compressors on the few shot sequence classification task. This calls into question the hypothesis that accuracy of NCD-based methods is predictable solely from compression rates.

drastically varying rates, the test accuracy spread is roughly the same, even with neural compressors with much lower compression rates.

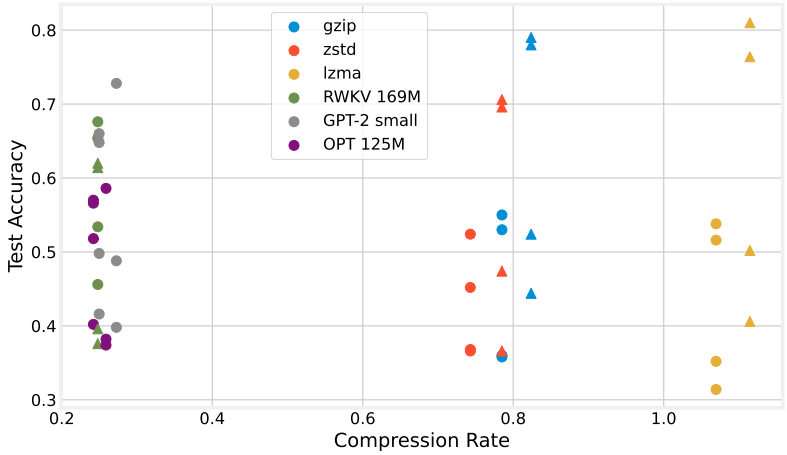


Figure 2: Test accuracy plotted against compression rate (lower is better compression) for AGNews and DBpedia across different few shot settings. Different shapes indicate different datasets, and each compressor is its own color. If compression rate and predictive performance were correlated, we would expect a diagonal relationship to occur, but none exists.

The internal workings of each compression algorithm are important in determining NCD quality. For example, *gzip* searches for repeated byte sequences only within a 32 KB sliding window. Since the concatenation of two inputs is a crucial component to calculating the NCD, large sequence lengths will prevent *gzip* from exploiting far-reaching redundancies between the two. Differences in neural network architectures among neural compressors may yield different abilities to identify these long range similarities and thus varying NCD quality.

### 4.2 Using other models for neural compression

In order to investigate the effects of the particular choice of model and model architecture on Neural NCD performance, we run identical experiments as above, swapping out RWKV 169M for two other models: GPT-2 117M (GPT-2 small) [26] and OPT 125M [27], as shown in Figure 3. Here we only use the *gzip* results out of the traditional compressors since they perform comparably and to aid visual clarity. We can see that all neural models outperform *gzip* in the same way on AGNews, and underperform it in the same way on DBpedia. Compression rates across models are shown in Table 2.

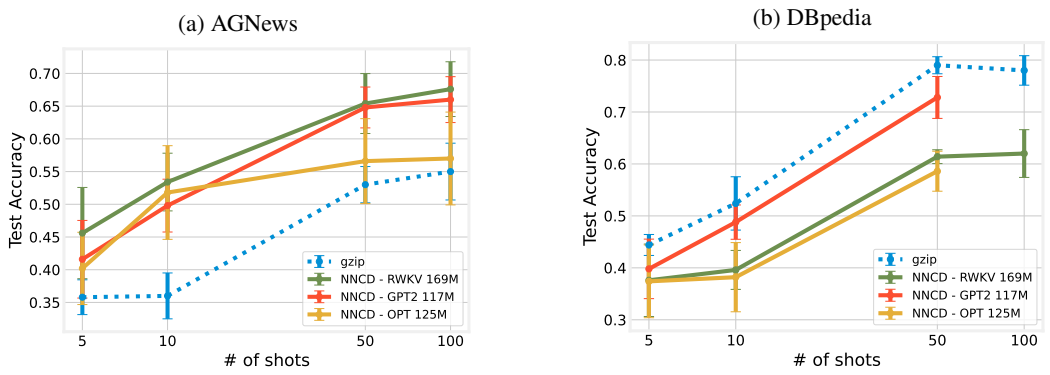


Figure 3: Comparison of RWKV 169M, GPT-2 117M, and OPT 125M, as the neural compressors used for Neural NCD. Neural compressors outperform *gzip* similarly on AGNews, and underperform it similarly on DBpedia.

### 4.3 Comparison with Euclidean distance between Latent Representations

For each neural compressor we test, we perform identical experiments as previously mentioned using the model's latent representation of a sequence and $k$NN with Euclidean distance. The results are shown in Figure 4. We find that depending on the choice of neural network model used for neural compression, Neural NCD can either outperform or underperform the Euclidean distance-based approach on sequence latent representations. For example, a Neural NCD approach with GPT-2 small outperforms using the model's latent representations, however with OPT 125M, latent representations drastically outperform a Neural NCD approach. This indicates that despite various LLMs providing similar compression ability, the usefulness of their latent representations for Euclidean distance-based approaches can differ.

Table 2: Compression rates when using various pretrained language models for neural compression on the AGNews and DBpedia datasets. For neural compressors, we use the raw compression rate which ignores the size of the neural network.

| Compressor | AGNews | DBpedia |
|---|---|---|
| gzip | 0.785 | 0.824 |
| RWKV 169M | 0.248 | 0.248 |
| GPT-2 117M | 0.250 | 0.273 |
| OPT 125M | 0.242 | 0.259 |

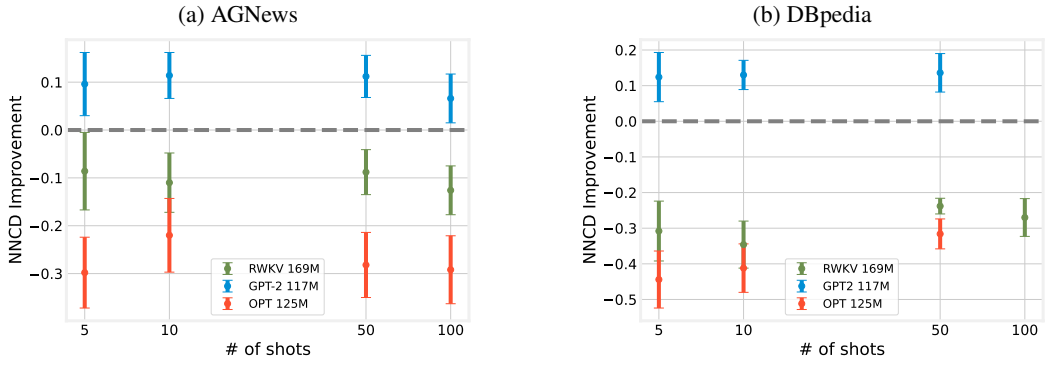


Figure 4: Test accuracy difference when comparing Neural NCD to Euclidean distance on sequence latent representations with 95% confidence interval. Values above 0 indicate Neural NCD outperforming Euclidean distance. For RWKV, the representation is the final hidden state. For GPT2 and OPT, we average the latent representation of each token. Despite comparable compression rates of each model, the quality and usefulness of distance between latent representations is highly variable across models.

## 5 Conclusion

We have shown that compression rate alone does not reliably predict NCD-based classification accuracy when using neural compressors in the text domain. With a variety of neural network architectures with which the machine learning community trains language models and neural compressors, it is likely that these architectural differences will also yield varying ability to exploit compressible redundancies across concatenated sequences yielding varying NCD quality. We've shown that neural models with different architectures and pretraining details tend to overperform or underperform traditional compressors in similar ways on the same dataset. Finally, we compared Neural NCD with latent representations from the same model, showing that Neural NCD outperforms latent representations for some models and under-performs them for others.

## References


[1] David J. C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[2] Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Commun. ACM*, 30(6):520–540, jun 1987.

[3] Ming Li, Xin Chen, Xin Li, Bin Ma, and Paul MB Vitányi. The similarity metric. *IEEE Transactions on Information Theory*, 50(12):3250–3264, 2004.

[4] Eamonn Keogh, Stefano Lonardi, and Chotirat Ann Ratanamahatana. Towards parameter-free data mining. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '04, page 206–215, New York, NY, USA, 2004. Association for Computing Machinery.

[5] Zhiying Jiang, Matthew Yang, Mikhail Tsirlin, Raphael Tang, Yiqin Dai, and Jimmy Lin. "low-resource" text classification: A parameter-free classification method with compressors. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 6810–6828, Toronto, Canada, July 2023. Association for Computational Linguistics.

[6] Zhiying Jiang, Yiqin Dai, Ji Xin, Ming Li, and Jimmy Lin. Few-Shot Non-Parametric Learning with Deep Latent Variable Model. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 26448–26461. Curran Associates, Inc., 2022.

[7] Fabrice Bellard. Nncp v2: Lossless data compression with transformer. `https://bellard.org/nncp/nncp_v2.1.pdf`, 2021.

[8] Byron Knoll. CMIX version 21. `http://www.byronknoll.com/cmix.html`, 2024.

[9] Byron Knoll. tensorflow-compress. `https://github.com/byronknoll/tensorflow-compress`, 2022.

[10] Matt Mahoney. Large text compression benchmark. `https://www.mattmahoney.net/dc/text.html`.

[11] Andrei N. Kolmogorov. On tables of random numbers. In *Proceedings of the Symposium on the Theory of Numbers*, pages 369–376, 1963.

[12] Chandra Shekhara Kaushik Valmeekam, Krishna Narayanan, Dileep Kalathil, Jean-Francois Chamberland, and Srinivas Shakkottai. Llmzip: Lossless text compression using large language models, 2023.

[13] Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression, 2024.

[14] Fabrice Bellard. ts_zip: Text compression using large language models. `https://bellard.org/ts_zip/`, 2024.

[15] Rebecca Saul, Mohammad Mahmudul Alam, John Hurwitz, Edward Raff, Tim Oates, and James Holt. Lempel-ziv networks. In *Proceedings on "I Can't Believe It's Not Better! - Understanding Deep Learning Through Empirical Falsification" at NeurIPS 2022 Workshops*, volume 187 of *Proceedings of Machine Learning Research*, pages 1–11. PMLR, 03 Dec 2023.

[16] Manuel Cebrián, Manuel Alfonseca, and Alfonso Ortega. Common pitfalls using the normalized compression distance: What to watch out for in a compressor. *Communications in Information & Systems*, 5(4):367–384, 2005.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[18] Edward Raff and Charles Nicholas. An Alternative to NCD for Large Sequences, Lempel-Ziv Jaccard Distance. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '17*, pages 1007–1015, New York, New York, USA, 2017. ACM Press.

[19] Edward Raff, Charles Nicholas, and Mark McLean. A New Burrows Wheeler Transform Markov Distance. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pages 5444–5453, 2020.

[20] Rebecca Schuller Borbely. On normalized compression distance and large malware: Towards a useful definition of normalized compression distance for the classification of large files. *Journal of Computer Virology and Hacking Techniques*, 12(4):235–242, December 2015.

[21] Stephanie Wehner. Analyzing worms and network traffic using compression. *J. Comput. Secur.*, 15(3):303–320, August 2007.

[22] João S. Resende, Rolando Martins, and Luís Antunes. A survey on using kolmogorov complexity in cybersecurity. *Entropy*, 21(12):1196, December 2019.

[23] Edward Raff and Charles K. Nicholas. Lempel-Ziv Jaccard Distance, an effective alternative to ssdeep and sdhash. *Digital Investigation*, feb 2018.

[24] Edward Raff and Charles Nicholas. Malware Classification and Class Imbalance via Stochastic Hashed LZJD. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, AISec '17, pages 111–120, New York, NY, USA, 2017. ACM.

[25] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. Rwkv: Reinventing rnns for the transformer era, 2023.

[26] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[27] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022.

# A Experiment Details

We use the labeled datasets AGNews, 20News, and DBpedia. For AGNews and DBpedia, we use the original dataset and classification setting: four classes for AGNews and fourteen classes for DBpedia. For 20News, we reduce the number of classes to two using only *alt.atheism* and *comp.graphics*. For $k$NN classification algorithm, we use $k = 3$ for all experiments.