# An Empirical Study of the Downstream Reliability of Pre-Trained Word Embeddings

**Anthony Rios**[1] **and Brandon Lwowski**[2]
Department of Information Systems and Cyber Security
University of Texas at San Antonio
San Antonio, TX 78249, USA
[1]anthony.rios@utsa.edu and [2]oek437@my.utsa.edu

**Abstract**

While pre-trained word embeddings have been shown to improve the performance of downstream tasks, many questions remain regarding their reliability: Do the same pre-trained word embeddings result in the best performance with slight changes to the training data? Do the same pre-trained embeddings perform well with multiple neural network architectures? What is the relation between downstream fairness of different architectures and pre-trained embeddings? In this paper, we introduce two new metrics to understand the downstream reliability of word embeddings. We find that downstream reliability of word embeddings depends on multiple factors, including, the handling of out-of-vocabulary words and whether the embeddings are fine-tuned.

## 1 Introduction

Pre-trained word embeddings have been shown to improve neural networks' performance across a wide variety of tasks. For instance, pretrained word embeddings improve the performance of models for text classification (Kim, 2014), relation extraction (Nguyen and Grishman, 2015), named entity recognition (Lample et al., 2016), and machine translation (Qi et al., 2018). However, neural network performance is unstable when retrained multiple times on the same dataset (Fard et al., 2016). Small changes in the training data can result in substantial differences in overall performance. Similarly, after retraining word embeddings instead of the model (i.e., to incorporate out-of-vocabulary words or capture changes in their semantic meanings), the instability of the word embeddings themselves, can cause differences in downstream performance (Leszczynski et al., 2020). Yet, retraining is still important. Otherwise, performance will deteriorate over time (Kim et al., 2017). In this paper, we expand on prior work (Leszczynski et al., 2020) to understand the downstream reliability of pre-trained word embeddings.

Leszczynski et al. (2020) discuss how model instability can increase a company's cost of deploying and keeping natural language processing (NLP) pipelines in production. Let $\mathbf{X}$ and $\bar{\mathbf{X}}$ be two different sets of embeddings, and let $g_{\mathbf{X}}$ and $g_{\bar{\mathbf{X}}}$ be two models trained on $\mathbf{X}$ and $\bar{\mathbf{X}}$. Leszczynski et al. (2020) define instability as

$$DI_T(\mathbf{X}, \bar{\mathbf{X}}) = \frac{1}{N}\sum_{i=1}^{N} \mathcal{L}(g_{\mathbf{X}}(z_i), g_{\bar{\mathbf{X}}}(z_i)), \tag{1}$$

where $\{z_i\}_{i=1}^{N}$ is a held-out test set for task some task $T$, and $\mathcal{L}$ is a fixed loss function. If $\mathcal{L}$ is the zero-one loss, then the measure captures the fraction of predictions that disagree between models trained on each set of embeddings. They analyze the model disagreement between two sets of embeddings trained on corpora collected from the same source at different time periods. They show that slight changes made to the beginning of an NLP pipeline (word embeddings) have a considerable impact on downstream performance. Intuitively, NLP engineers can potentially spend a lot of time searching for potential bugs caused by concept drift in word embeddings. Moreover, the time and resources required to train many state-of-the-art neural networks are growing substantially over time (Strubell et al., 2019; Schwartz et al., 2019). This trend results in a potential increase in greenhouse gasses (Strubell et al., 2019), as well as increasing the social cost to participate in natural language processing research (Schwartz et al., 2019).

In many scenarios, it is expected that different sets of embeddings may result in better performance, e.g., training on new domains (Moen and Ananiadou, 2013). Thus, there are many aspects related to the reliability and stability of the NLP pipeline that is neither cost-effective nor efficient and have yet to be studied. For example, instead of slight changes in word embeddings trained on the same source corpora, but collected at different times, What if task $\mathcal{T}$'s training dataset changed? What if the embeddings are the same, and the model $g()$ changes? Will the same pre-trained embeddings work if the dataset changes? Will the same embeddings work for different models? How will changes in pre-trained embeddings affect downstream fairness? Overall, if an NLP engineer does not need to evaluate every model, with every set of pre-trained word embeddings, for every question of interest, both the computational cost of putting models into production and the engineer's time spent evaluating model variations may be reduced.

Toward addressing the above methodological gaps, this paper presents the following contributions: **(1.)** We introduce two new metrics to measure the downstream reliability of word embeddings: Cross–Dataset Reliability (CDR) and Cross-Model Reliability (CMR); **(2.)** using our reliability measures, we provide a comprehensive analysis of the downstream reliability of word embeddings on three offensive language datasets, multiple standard classification and fairness metrics, and multiple neural network architectures; and **(3.)** we provide an in-depth discussion about our findings, their implications, as well as the limitations of our study.

## 2 Related Work

In this section, we briefly describe the main areas of research in which this paper is based: word embeddings, word embedding instability, and the downstream impact of word embeddings.

**Word Embeddings.** Word embeddings capture the distributional nature between words, i.e., word vectors will encode the context in which words frequently appear. There are multiple word embedding methods, such as, latent semantic analysis (Deerwester et al., 1990), Word2Vec (Mikolov et al., 2013a; Mikolov et al., 2013b), and GLOVE (Pennington et al., 2014). The basic component driving our research is that pre-trained word embeddings have been shown to be useful for a wide variety of downstream NLP tasks, such as text classification (Kim, 2014), relation extraction (Nguyen and Grishman, 2015), named entity recognition (Lample et al., 2016), and machine translation (Qi et al., 2018). So, we want to understand more about their downstream impact.

**Word Embedding Instability.** There have been significant findings in the instability of word embeddings (Hellrich and Hahn, 2016; Hellrich et al., 2019; Antoniak and Mimno, 2018; Burdick et al., 2018; Pierrejean and Tanguy, 2018). Hellrich and Hahn (2016) show that when investigating neighboring words in the embedding space, there is low reliability in which words are surrounding a particular token across multiple runs. In fact, stability is only present when word vectors are trained on a small corpus with a limited vocabulary (Hellrich and Hahn, 2016). Generally, even the smallest change in the training corpus causes high variation in the nearest neighbor distances. Furthermore, these variations are not only subject to low frequency words, instability is found to be present in vocabulary words that occur relatively frequently (Hellrich and Hahn, 2016). Instability has been shown in multiple algorithms (e.g., Skip-Gram (Hellrich and Hahn, 2016) and SVD (Hellrich et al., 2019)), as well as in different training corpora (e.g., historical text (Hellrich and Hahn, 2016) and social media (Antoniak and Mimno, 2018)).

**Instability and the Downstream Impact of Word Embeddings.** Rogers et al. (2018) explored morphological, semantic, and distributional factors that are correlated with downstream performance. Their work pointed to ways of improving the downstream performance of neural architectures by modifying word embeddings. Yet, while understanding what results in better performing embeddings are important, the instability of machine learning methods after small changes to features, word embeddings, and training data on the the downstream performance of certain tasks is troubling (Fard et al., 2016; May et al., 2019; Leszczynski et al., 2020). If the data distribution changes rapidly, then the machine learning models need to be retrained frequently, or at least, the word embeddings need to be retrained to handle out-of-vocabulary words and their respective meanings. Burdick et al. (2018) study which factors correlate with word embedding stability. A few of their findings include, GLOVE (Pennington et al., 2014) is one of the most stable methods, and part-of-speech is one of the biggest factors of instability. Fard et al. (2016) analyze the instability of machine learning models. They provide a metric of model instability, called prediction churn, and provide a markov chain monte carlo techinques to reduce churn, i.e., to reduce predictions changing dramatically. May et al. (2019) provides insight into correlationsbetween downstream performance and the compression abilities of word embeddings. Leszczynski et al. (2020) provide the first metric for downstream instability of word embeddings, and they show that increasing the embedding dimensions can reduce embedding instability. This paper builds on the work of Leszczynski et al. (2020) by providing insight into practical downstream instability issues of word embeddings.

## 3 Methodology

In this paper, we use two definitions of reliability: cross-dataset reliability (CDR) and cross-model reliability (CMR). These definitions differ from the definition of word embedding stability defined by Leszczynski et al. (2020) (as shown in Equation 1). Formally, let $\mathcal{Z} = \{z_i\}_{i=1}^{N}$ and $\bar{\mathcal{Z}} = \{\bar{z}_i\}_{i=1}^{\bar{N}}$ represent two independent datasets. $f(\mathcal{Z}, m) \in \mathbb{R}^q$ is a vector of evaluation metrics (e.g., AUC, F1, accuracy, etc.) for model $m$ (e.g., CNN) on dataset $\mathcal{Z}$. The scores in $f(\mathcal{Z}, m)$ are the result of training model $m$ using $q$ different pre-trained word embeddings. Therefore, $f(\mathcal{Z}, m)_k$ is the result (e.g., AUC) of model $m$ trained on dataset $\mathcal{Z}$ with pre-trained embeddings $k$. Thus, we define CDR as

$$CDR(\mathcal{Z}, \bar{\mathcal{Z}}) = \mathcal{C}(f(\mathcal{Z}, m), f(\bar{\mathcal{Z}}, m)), \tag{2}$$

where $\mathcal{C}()$ represents the *Spearman rho* correlation between vectors $f(\mathcal{Z}, m)$ and $f(\bar{\mathcal{Z}}, m)$. Intuitively, a high correlation means that the word embeddings which result in the best and worst performance for model $m$ are similar for dataset $\mathcal{Z}$ and dataset $\bar{\mathcal{Z}}$. Next, CMR is defined as

$$CMR(m, \bar{m}) = \mathcal{C}(f(\mathcal{Z}, m), f(\mathcal{Z}, \bar{m})), \tag{3}$$

where $\bar{m}$ and $m$ are two different models such as an LSTM and CNN, respectively. Contrary to CDR, the intuition behind CMR is that a high correlation value means that the embeddings which result in the best and worst performance on datasets $\mathcal{Z}$ for model $m$ are similar to the results of model $\bar{m}$ on dataset $\mathcal{Z}$. To improve the robustness of our measurements, each result $f(\mathcal{Z}, m)_k$ is the average of training model $m$ on dataset $\mathcal{Z}$ ten times using the $k$-th set of embeddings with different random seeds.

### 3.1 Datasets

We use three English offensive language datasets in this paper: a sexist dataset (Sexist), an abusive dataset (Abusive), and a general offensive language dataset (OLID). Essentially, each dataset contains offensive language. Sexist language is a subset of abusive language (Waseem and Hovy, 2016), and abusive language is a subset of offensive language (Zampieri et al., 2019). Therefore, classifiers trained to detect general offensive language should detect both abusive and sexist tweets. The datasets were chosen because they differ slightly, but, their over-arching theme is the same. This similarity is discussed for the Sexist and Abusive datasets in Park et al. (2018). Moreover, for each dataset, we use 60% for training, 20% for validation, and 20% for testing. Each dataset's test split is used to calculate the reliability metrics defined in Equations 2 and 3, as $\mathcal{Z}$ and $\bar{\mathcal{Z}}$. We briefly describe each dataset below:

**Sexist Tweets Dataset (Sexist).** The Sexist dataset is a collection of tweets that was annotated as 'Sexist', 'Racist', or 'Neither'. It was originally collected and used in Waseem and Hovy (2016) and Waseem (2016). It was then used to evaluate the fairness of offensive language classifiers in Park et al. (2018). The tweets were collected by searching for words related to sexism, then, using criteria from critical race theory, the tweets were manually annotated by experts. Following Park et al. (2018), we focus on 'Sexist' tweets by removing 'Racist' tweets from the dataset. The dataset contains 14,937 total tweets, of which 3,378 are annotated with the 'Sexist' class.

**Abusive Language Dataset (Abusive).** The Abusive dataset contains crowdsourced tweets labeled with one of four classes: 'None', 'Spam', 'Abusive', and 'Hateful'. Following Park et al. (2018), we combine 'None'/'Spam' together, and 'Abusive'/'Hateful' together. This combination results in 99,800 total tweets, of which 31,985 are categorized as 'Abusive'.

**Offensive Language Dataset (OLID).** The OLID dataset (Zampieri et al., 2019) contains 14,100 tweets labeled using a hierarchical annotation scheme where the top level (task A) differentiates 'Offensive' and 'Not Offensive' tweets. The bottom level (task C) categorizes insults/threats as targeting an individual, group, or other. For the purposes of this paper, we only use the first level, task A, of the hierarchy. The first level contains two classes: 'Offensive' and 'Not Offensive'. This results in a total of 4,640 tweets categorized as 'Offensive'.

### 3.2 Word Embeddings and Imputation Strategies

We test 17 publicly available sets of embeddings that vary in terms of the source embedding algorithm, training data, and dimension. The embeddings include GLOVE (Pennington et al., 2014), FastText (Bojanowski et al., 2016), and SkipGram (Mikolov et al., 2013b) variations. The complete listing of the embeddings used in our experiments can be found in the Supplementary Material.

When using pre-trained word embeddings, there are implementation-level details that can affect downstream stability and reliability. For example, how should out-of-vocabulary words (i.e, words in the training dataset, but not in the pre-trained embeddings) be handled? Should unknown words be ignored, or should the unknown words be initialized randomly? Furthermore, the choice between fine-tuning the embeddings or keeping them static can also impact downstream performance and reliability.

Overall, we make use of three imputation strategies in our experiments: "*Impute*", "*No Impute*", and "*Static*". For "*Impute*", we initialize the embeddings of words missing in the pretrained embedding set, but that appear in the training dataset, with a random embedding using a uniform distribution in the range from -.1 to .1. The embeddings learned using the "*Impute* " strategy are fine-tuned during training. For the "*No Impute*" strategy, out-of-vocabulary words are ignored, but the embeddings are still fine-tuned during training. Finally, the "*Static*" strategy ignores out-of-vocabulary words, and the pre-trained embeddings are static during training, i.e., the embeddings are not fine-tuned.

### 3.3 Text Classification Models

In our experiments, we explore four neural network architectures: Convolutional Neural Networks (CNN), Neural Bag-of-Words (NBoW), Long Short-Term Memory Networks (LSTM), and Gated Recurrent Units (GRU). Every model uses standard word embeddings as their input (i.e., contextual embeddings are not tested).

**Covolutional Neural Networks (CNN).** We use the model proposed by Kim (2014). Essentially, the model is a shallow CNN with max-over-time pooling, followed by a sigmoid output layer. Let $\mathbf{x}_i \in \mathbb{R}^d$ represent a $d$-dimensional embedding of the $i$-th word in a document. The CNN learns to extract ngrams from text that are predictive of the downstream task. Formally, each span of $s$ words are concatenated, $[\mathbf{x}_{i-s+1}; \ldots; \mathbf{x}_i]$, into a contextual vector $\mathbf{c}_j \in \mathbb{R}^{s(d+2e)}$. Next, using a rectified linear unit (Nair and Hinton, 2010) $f()$, the covolution operation is applied to each vector,

$$\hat{\mathbf{c}}_j = f(\mathbf{W}\mathbf{c_j} + \mathbf{b}),$$

where $\mathbf{b} \in \mathbb{R}^q$. Next, given the convolved context vectors $[\hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, \ldots, \hat{\mathbf{c}}_{n+s-1}]$, the CNN map them into a fixed sized vector using max-over-time pooling

$$\mathbf{g} = [\hat{c}^1_{max}, \hat{c}^2_{max}, \ldots, \hat{c}^q_{max}], \quad \text{where} \quad \hat{c}^j_{max} = max(\hat{c}^j_1, \hat{c}^j_2, \ldots, \hat{c}^j_{n+s-1}),$$

such that $\hat{c}^j_{max}$ represents the max value across the $j$-th feature map. Finally, $\mathbf{g}$ is passed to a sigmoid output layer. We train the CNN with filter sizes that span three, four, and five words. Furthermore, we use a total of 300 filters for each size, and the Adam optimizer (Kingma and Ba, 2014).

**Neural Bag-of-Words (NBoW).** Unlike the CNN, which learns to extract predictive ngrams from text, the NBoW model only processes unigrams. Specifically, NBoW sums the word embeddings

$$\mathbf{h} = \sum_{i=1}^{Q} \mathbf{x}_i$$

where $Q$ is the total number of words in an instance (e.g., a tweet). The summed vector $\mathbf{h}$ is passed to a sigmoid output layer. We train the NBoW model with the Adam optimizer (Kingma and Ba, 2014).

**Long Short-Term Memory Networks (LSTM).** While CNNs only extract informative n-grams from text, recurrent neural networks (RNNs) are able to capture long term dependencies between words. For our RNN method, we use long-short-term-memory (LSTM) (Gers et al., 2000), specifically we use a variant introduced by Graves (2012),

$$\begin{aligned}
\mathbf{i}_i &= sigmoid(\mathbf{x}_i\mathbf{W}_i + \mathbf{b}_i + \mathbf{h}_{i-1}\mathbf{U}_i)\\
\mathbf{f}_i &= sigmoid(\mathbf{x}_i\mathbf{W}_f + \mathbf{b}_f + \mathbf{h}_{i-1}\mathbf{U}_f),\\
\mathbf{o}_i &= sigmoid(\mathbf{x}_i\mathbf{W}_o + \mathbf{b}_o + \mathbf{h}_{i-1}\mathbf{U}_o),\\
\mathbf{p}_i &= tanh(\mathbf{x}_i\mathbf{W}_c + \mathbf{b}_c + \mathbf{h}_{i-1}\mathbf{U}_c),\\
\mathbf{m}_i &= \mathbf{f}_i * \mathbf{m}_{i-1} + \mathbf{i}_i * \mathbf{p}_i,\\
\mathbf{h}_i &= \mathbf{o}_i * tanh(\mathbf{m}_i),
\end{aligned}$$

where $\mathbf{i}_i$, $\mathbf{f}_i$, $\mathbf{o}_i$ represent the input, forget, and output gates. The hidden state vector $\mathbf{h}_Q$ of the final word in each sentence is passed to a sigmoid output layer. The LSTM model is trained with hidden state size of 512 using the Adam optimizer (Kingma and Ba, 2014).

**Gated Recurrent Units (GRU).** We also explore a variant of the LSTM architecture, GRUs (Cho et al., 2014). GRUs are similar to LSTMs, but they have fewer parameters and do not have an output gate. The GRU we use is defined as

$$\begin{aligned}
\mathbf{z}_i &= sigmoid(\mathbf{x}_i\mathbf{W}_z + \mathbf{b}_z + \mathbf{h}_{i-1}\mathbf{U}_z)\\
\mathbf{f}_i &= sigmoid(\mathbf{x}_i\mathbf{W}_f + \mathbf{b}_f + \mathbf{h}_{i-1}\mathbf{U}_f),\\
\hat{\mathbf{h}}_i &= tanh(\mathbf{x}_i\mathbf{W}_h + \mathbf{b}_h + (\mathbf{h}_{i-1} \odot \mathbf{f}_i)\mathbf{U}_h),\\
\mathbf{h}_i &= (1 - \mathbf{z}_i) \odot \mathbf{h}_{i-1} + \mathbf{z}_i \odot \hat{\mathbf{h}}_i,
\end{aligned}$$

where $\mathbf{h}_i$, $\hat{\mathbf{h}}_i$, $\mathbf{z_i}$, and $\mathbf{f}_i$ are the output, candidate activation, update gate, and forget gate vectors, respectively. $\odot$ denotes the Hadamard product, i.e., element-wise multiplication. Like the LSTM, the final output vector $\mathbf{h}_Q$ is passed to a full-connected sigmoid output layer. We train the GRU model with a hidden state size of 512 using the Adam optimizer (Kingma and Ba, 2014).

## 4 Experiments

In this section, we describe the evaluation metrics used in our experiments, and relate the overall performance of the models and imputation strategies to reliability.

### 4.1 Evaluation Metrics

We use four evaluation metrics in our experiments: AUC, GAUC, FPED, and FNED. AUC is the standard area under the receiver operating characteristic curve (ROC). This metric is used to evaluate the performance of the model on a held-out test set. GAUC is also the area under the ROC curve. However, instead of evaluating on the held-out test set, the AUC is calculated on the gender bias template (GenTemp) dataset, which we describe below. The FPED and FNED scores are also computed on GenTemp. Overall, each metric is used to evaluate reliability as part of Equations 2 and 3 (i.e., they are the output of $f(\mathcal{Z}, m)$).

While many metrics have been proposed to evaluate fairness (Zliobaite, 2015; Hardt et al., 2016; Dixon et al., 2018a; Borkan et al., 2019; Beutel et al., 2019; Mitchell et al., 2019), unfortunately, most methodologies require ground-truth or inferred demographic annotations (Badjatiya et al., 2019; Garg et al., 2018). In the absence of annotated demographic data, (Dixon et al., 2018a) propose fuzzing methods—a method of testing fairness with simulated data to analyze how predictions change if the topic of the tweet stays the same, but the text in pre-defined templates (e.g. I am {adjective}) is slightly

| | **Sexist** | | | | **Abusive** | | | | **OLID** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **AUC ↑** | **GAUC ↑** | **FPED ↓** | **FNED ↓** | **AUC ↑** | **GAUC ↑** | **FPED ↓** | **FNED ↓** | **AUC ↑** | **GAUC ↑** | **FPED ↓** | **FNED ↓** |
| | | | | | | **Impute Embedding Strategy** | | | | | | |
| **CNN** | .917 | .556 | .099 | .158 | .885 | **.880** | .014 | .017 | .830 | .897 | .065 | .117 |
| **NBoW** | .916 | .552 | .108 | .214 | .882 | .764 | .012 | **.016** | .803 | .708 | .133 | .143 |
| **LSTM** | **.934** | .552 | .129 | .203 | .890 | .834 | .012 | .022 | .859 | .918 | .079 | .152 |
| **GRU** | **.934** | .545 | **.075** | .150 | .891 | .789 | **.009** | .028 | **.861** | **.922** | .071 | .083 |
| **AVG** | **.925** | .551 | **.103** | .181 | **.887** | **.816** | **.012** | **.021** | .838 | **.861** | .087 | .124 |
| | | | | | | **No Impute Embedding Strategy** | | | | | | |
| **CNN** | .873 | .566 | .203 | .220 | .875 | .572 | .210 | .221 | .832 | .893 | .011 | .034 |
| **NBoW** | .874 | .551 | .132 | .231 | .880 | .737 | .013 | .015 | .804 | .698 | **.007** | .052 |
| **LSTM** | .889 | **.571** | .333 | .300 | **.894** | .816 | .011 | .017 | .859 | .911 | .013 | .035 |
| **GRU** | .889 | **.573** | .275 | .266 | **.894** | .796 | .010 | .017 | **.862** | .913 | .014 | .027 |
| **AVG** | .881 | **.565** | .236 | .254 | .886 | .730 | .061 | .067 | **.839** | .854 | .015 | .037 |
| | | | | | | **Static Embedding Strategy** | | | | | | |
| **CNN** | .841 | .524 | **.078** | **.104** | .845 | .522 | .083 | .097 | .796 | .853 | .010 | **.024** |
| **NBoW** | .792 | **.577** | .151 | .158 | .803 | .687 | .025 | .030 | .759 | .682 | .015 | .062 |
| **LSTM** | .862 | .539 | .172 | .198 | .887 | .834 | .022 | .021 | .829 | .853 | **.009** | **.025** |
| **GRU** | .863 | .530 | .136 | .153 | .887 | .808 | .020 | .022 | .833 | .870 | **.009** | **.025** |
| **AVG** | .840 | .542 | .135 | **.153** | .856 | .713 | .037 | .043 | .804 | .815 | **.012** | **.034** |

Table 1: Overall Performance of each model (CNN, BoW, LSTM, and GRU) on all three datasets (Sexist, Abusive, and OLID) averaged over all pre-trained word embeddings. ↑ and ↓ mark whether the performance is better with a higher (↑) or lower (↓) score. Scores are reported on a held-out split of each dataset (AUC) as well as the synthetic gender dataset (GAUC, FPED, and FNED).

altered. For example, fuzzing techniques will randomly change demographic words (e.g., "he", "she", "husband", and "wife") in a tweet without changing its meaning.

In this paper we use the code released by Dixon et al. (2018b),[1]. Specifically, we generated 1,424 samples (712 pairs) by filling the templates with common gender identity pairs (e.g., male/female, man/woman, etc.). We call this set of filled templates GenTemp. The created templates contain neutral and offensive nouns and adjectives inside the vocabulary to retain balance in 'Not Offensive' and 'Offensive' samples. See the Supplementary Material for a complete listing of the nouns and adjectives.

Following the experimental setup of Park et al. (2018), to measure the fairness of the different models, we use the AUC metric on GenTemp (GAUC), and compare the absolute differences between the false positive rate and false negative rate calculated independently for each gender. False-positive (FPR) and false-negative rates (FNR) are defined as

$$FPR = \frac{FP}{FP + TN} \qquad \text{and} \qquad FNR = \frac{FN}{FN + TP}$$

where TP, FP, FN, and TN represent the number of true positives, false positives, false negatives, and true negatives, respectively. Moreover, we use the False Positive Equality Difference (FPED) and False Negative Equality Difference (FNED) (Dixon et al., 2018a). FPED and FNED are defined as

$$\text{FPED} = \sum_{t \in T} |FPR - FPR_t| \qquad \text{and} \qquad \text{FNED} = \sum_{t \in T} |FNR - FNR_t|,$$

respectively, where $T = \{\text{Male}, \text{Female}\}$. FPR and FNR represent the overall false positive and false negative rates, respectively. $FPR_t$ and $FNR_t$ represent the group-specific (i.e., Male or Female) false positive and false negative rates.

## 4.2 Overall Performance

We report the overall performance of each model on the three offensive language datasets in Table 1. While the goal of this paper is not to develop the best offensive language classifiers, it is important to

[1]`https://github.com/conversationai/unintended-ml-bias-analysis`

| | **Sexist ⇔ Abusive** | | | | **Sexist ⇔ OLID** | | | | **Abusive ⇔ OLID** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **AUC ↑** | **GAUC ↑** | **FPED ↑** | **FNED ↑** | **AUC ↑** | **GAUC ↑** | **FPED ↑** | **FNED ↑** | **AUC ↑** | **GAUC ↑** | **FPED ↑** | **FNED ↑** |
| **Impute Embedding Strategy** | | | | | | | | | | | | |
| **CNN** | -.085 | .388 | .368 | -.050 | .300 | .756 | .553 | -.182 | .700 | .529 | .746 | .032 |
| **NBoW** | .247 | -.550 | -.541 | -.003 | .650 | .788 | .362 | .159 | .653 | -.685 | -.503 | -.162 |
| **LSTM** | .741 | .476 | -.329 | .147 | .694 | .365 | .271 | -.056 | .741 | .612 | .200 | .456 |
| **GRU** | .759 | .594 | .194 | -.500 | .447 | .341 | .274 | -.168 | .685 | .562 | .329 | .426 |
| **AVG** | .415 | .227 | -.077 | -.101 | .523 | .563 | .365 | -.062 | .695 | .254 | .193 | .188 |
| **No Impute Embedding Strategy** | | | | | | | | | | | | |
| **CNN** | .885 | .899 | .791 | .565 | .626 | .496 | .174 | -.538 | .591 | .591 | .162 | -.412 |
| **NBoW** | .645 | .056 | -.302 | -.153 | .765 | .835 | .097 | .132 | .449 | -.162 | -.071 | .000 |
| **LSTM** | .897 | .679 | -.418 | .129 | .853 | .668 | -.197 | -.416 | .938 | .724 | .374 | .188 |
| **GRU** | .938 | .503 | -.709 | -.365 | .821 | .644 | -.179 | -.424 | .894 | .588 | .376 | .300 |
| **AVG** | .841 | .534 | -.159 | .044 | .766 | .661 | -.026 | -.311 | .718 | .435 | .210 | .019 |
| **Static Embedding Strategy** | | | | | | | | | | | | |
| **CNN** | **.938** | **.894** | **.971** | **.903** | **.909** | **.835** | -.524 | **.609** | **.885** | **.882** | -.500 | -.559 |
| **NBoW** | .779 | .741 | -.132 | .362 | .779 | .685 | .271 | .529 | .929 | .876 | .165 | .374 |
| **LSTM** | .897 | .612 | -.500 | -.003 | .876 | .879 | -.488 | -.759 | .982 | .691 | .708 | -.097 |
| **GRU** | .897 | .785 | -.662 | .188 | .859 | .894 | -.541 | -.888 | .947 | .821 | .509 | -.185 |
| **AVG** | **.878** | **.758** | -.081 | **.363** | **.856** | **.824** | -.321 | -.127 | **.936** | **.818** | **.220** | -.117 |

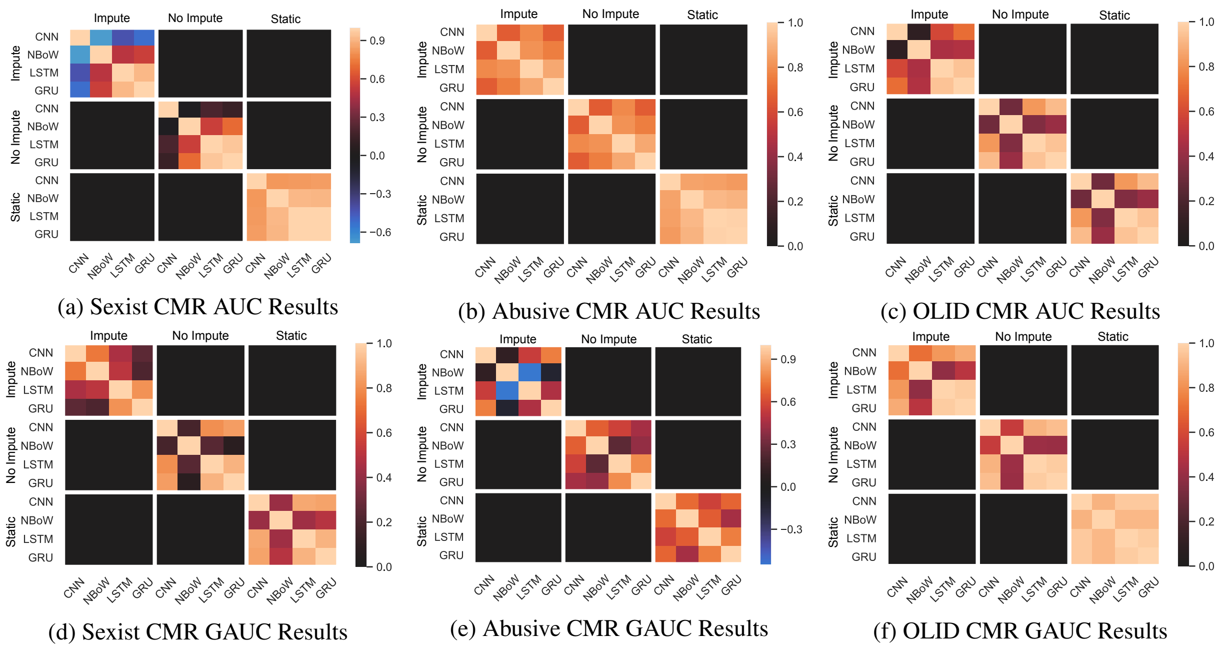
Table 2: Cross-dataset reliability (CDR) results measured by the *Spearman rho correlation* between the performance scores (AUC, GAUC, FPED, and FNED) of the same model trained on different datasets. The correlation measures whether the word embeddings that result in the best (worst) performance for a given model are the same on different, but similar, datasets. Higher (↑) scores marks more correlation.

understand each model's performance when analyzing reliability. Each model's performance in Table 1 is averaged over all 17 embeddings and the ten repeated runs of each model. Furthermore, the AVG rows mark the average performance of an embedding strategy across all four models. Concerning the AUC on each dataset's held-out test set, we find that the "*Impute*" and "*No Impute*" embedding strategies result in the best performance on average. For instance, The AVG AUC on the Sexist dataset is 0.925, which is more than 4% higher than the "*No Impute*" strategy (0.881 AUC) and 8% higher than the "*Static*" AVG (0.840). Thus, static embeddings generally result in sub-optimal performance. We find a similar pattern for GAUC—the AUC results on the gender bias template dataset. The GAUC for the Abusive dataset (0.816 GAUC) is nearly 9% better than the use of the "*No Impute*" strategy (0.730 GAUC). We also find that the LSTM and GRU models generally result in the best AUC and GAUC, e.g., on the Sexist dataset, the LSTM's AUC (0.934) is nearly 2% better than the CNN and BoW models.

Based on the FPED and FNED metrics, the fairest model and imputation strategies vary depending on the dataset. For instance, the "*Static*" embedding imputation strategy obtains the best scores for FPED (0.012) and FNED (0.034) on the OLID dataset, which is considerably better than the "*Impute*" strategy (FPED 0.087; FNED 0.124). Contrary to the OLID results, the "*Impute*" strategy on the Abusive dataset results in the best FPED (0.012) and FNED (0.021) scores, which is nearly 5% better than the "*No Impute*" Strategies FPED (0.061) and FNED (0.067). Overall, while the best model and imputation strategies vary for FPED and FNED, the bias is small in the Abusive and OLID datasets.

## 4.3 Cross-Dataset Reliability Analysis

The CDR results are shown in Table 2. **We find that the use of static embeddings results in stable cross-dataset AUC and GAUC performance**. For instance, the correlation between the AVG Impute model and Static models, on the Sexist and Abusive datasets, jumps from 0.415 to 0.878. Intuitively, this means if the word embeddings are static, then the embeddings that result in the best performance for a certain model (e.g., CNN) on the Sexist dataset will likely result in the best performance on the Abusive dataset. We find similar correlation improvements between Sexist ⇔ OLID (0.523 to 0.586) and Abusive ⇔ OLID (0.695 to 0.936). Additionally, in the static embedding setting, we see high correlation between



(a) Sexist CMR AUC Results (b) Abusive CMR AUC Results (c) OLID CMR AUC Results

(d) Sexist CMR GAUC Results (e) Abusive CMR GAUC Results (f) OLID CMR GAUC Results

Figure 1: Cross-Model Reliability (CMR) correlation heatmap measured by the *Spearman rho correlation* between the performance scores (AUC and GAUC) of different models trained on the same dataset. The correlation measures whether the word embeddings that result in the best (worst) performance for a given model are the same for different models.

datasets that differ substantially in size (e.g., Sexist ⇔ Abusive). This result suggests that small datasets could potentially be used to find which pre-trained embeddings work the best for a given problem, then the same embeddings will generalize to a larger dataset.

While the AUC and GAUC CDR correlations improve using static embeddings, the CDR scores for the FPED and FNED fairness metrics do not have any noticeable correlation improvements. For instance, with static embeddings, the AVG CDR FPED score between the Sexist and OLID datasets is -0.321, i.e., on average, the embeddings that result in the fairer models in the Sexist dataset are inversely related to the embeddings in OLID. We find similar patterns in the FPED results for Sexist ⇔ Abusive (-0.081) and FNED results on Abusive ⇔ OLID (-0.117) with static embeddings. The pattern continues for the "*Impute*" and "*No Impute*" embedding strategies.

### 4.4 Cross-Model Reliability Analysis

In Figure 1, we report correlation heatmaps of the CMR study on each dataset. Note that all the results are in the diagonal blocks to report cross-model correlation on the same dataset. We analyze two of the evaluation metrics: AUC and GAUC. Overall, similar to the CDR results, we find that static embeddings result in more downstream reliability. For example, in Figure 1a, we see a steady improvement in CMR from the "*Impute*" strategy—where some models are negatively correlated with each other (e.g., the CNN and LSTM)—to the "*Static*" embedding imputation strategy. We find similar results for the abusive and OLID datasets shown in Figures 1b and 1c, respectively. In addition, the CMS correlation between the GRU and LSTM models is consistently higher than other pairs of models. This result suggests that the more similarities between two neural network architectures, the more likely the embeddings that perform well for one, will perform well for the other.

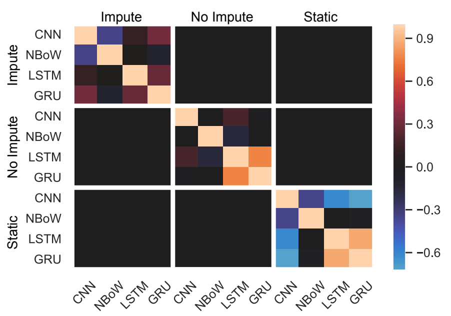


Figure 2: Abusive Cross-Model Reliability (CMR) FPED Results.

In Figure 2, we show the FPED CMR results on the Abusive dataset. Similar to the CDR findings, we find that there is little downstream fairness reliability of the word embeddings. Specifically, we find nearly zero correlation between the FPED results of the CNN and the LSTM models with the "*Impute*" strategy. Furthermore, with the "*Static*" embedding strategy, many of the CMR results have negative correlations. The results imply that while the static embeddings resulting in the best AUC and GAUC performance are similar across models, there is no guarantee about FPED and FNED performance. We have similar results for the FPED and FNED results for all three datasets. However, because of space limitations, the rest of the FPED and FNED Figures can be found in the Supplementary Material.

## 5 Discussion

We have two major findings in this paper. First, **if we fine-tune word embeddings, then the downstream performance will be erratic between models and datasets**. Unfortunately, as was shown in Section 4.2, fine-tuning is vital to achieving the most accurate model. Therefore, if overall performance is the primary goal, then every embedding-model combination must be explored. However, if computational efficiency is essential, and an NLP engineer wants to train a model on recently collected data. They only need to evaluate the best embeddings from their previous study. Likewise, if an NLP engineer wants to efficiently test a new model with static embeddings, based on the CMR results, the engineer only needs to evaluate the new model on the embeddings that initially resulted in the best performance.

Our second major finding is that, for the FPED and FNED fairness metrics, **the reliability of the downstream fairness of word embeddings is erratic across models, datasets, and embedding imputation strategies**. So, if the FPED and FNED fairness metrics are essential for the downstream task, it is crucial to test every embedding-model combination. However, many industries (e.g., health, government, etc.) that have applications where fairness is mission-critical may not have the resources for large-scale experimentation. Therefore, even if these industries rely on human-NLP collaboration, finding efficient and reliable methods of fairness estimation may be helpful, even if the accuracy does not necessarily achieve state-of-the-art performance. While there has been research about reducing the social cost of testing the fairness of text classification models (Rios, 2020), all embedding-model combinations still need to be tested in their framework. It is our opinion, that reducing the social and computational cost of testing the fairness of NLP methodologies is an important avenue of future research.

One of the major limitations of this study is that we treat gender as a binary concept while evaluating fairness. Gender is difficult (impossible) to detect automatically because gender is not a binary classification task. People may identify as binary trans people, non-binary people, or as gender non-conforming people. In this work we do not classify users into gender categories, but we do use a synthetic dataset to estimate binary gender fairness. As future work, we believe it is best to perform controlled experiments where we ask users how they identify, rather than grouping them automatically or using toy synthetic test sets. This approach—of asking rather than predicting—is also suggested for studies about gender in Scheuerman et al. (2019).

Finally, it is important to note the similarities and differences of our reliability metrics (CDR and CMR) to domain adaptation. Generally, domain adaptation methods attempt to improve the performance of machine learning models on data for a task that does not match the original data distribution (Ramponi and Plank, 2020). In this paper, we explore the downstream reliability of word embeddings when applied to datasets that do not match the original data distribution. Yet, the downstream performance of word embeddings may be reliable, but not generalize well in terms of overall performance. Ideally, future work should explore methods that generalize well and are reliable.

## 6 Conclusion

In this paper, we develop metrics of downstream reliability of pre-trained word embeddings. Specifically, we measure the downstream reliability of word embeddings across datasets and models. Our findings conclude that the performance of word embeddings are reliable when they are static, i.e., when they are not fine-tuned. This implies, without fine-tuning, that every publicly available set of word embeddings does not need to be evaluated if an NLP engineer trains on an updated dataset, or tests a different neural network architecture.

## References

Maria Antoniak and David Mimno. 2018. Evaluating the stability of embedding-based word similarities. *Transactions of the Association for Computational Linguistics*, 6:107–119.

Pinkesh Badjatiya, Manish Gupta, and Vasudeva Varma. 2019. Stereotypical bias removal for hate speech detection task using knowledge-based generalizations. In *The World Wide Web Conference*, pages 49–59.

Alex Beutel, Jilin Chen, Tulsee Doshi, Hai Qian, Allison Woodruff, Christine Luu, Pierre Kreitmann, Jonathan Bischof, and Ed H Chi. 2019. Putting fairness principles into practice: Challenges, metrics, and improvements. *arXiv preprint arXiv:1901.04562*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. *arXiv preprint arXiv:1903.04561*.

Laura Burdick, Jonathan K Kummerfeld, and Rada Mihalcea. 2018. Factors influencing the surprising instability of word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2092–2102.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018a. Measuring and mitigating unintended bias in text classification. In *Conference on AI, Ethics, and Society*.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018b. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 67–73.

Mahdi Milani Fard, Quentin Cormier, Kevin Canini, and Maya Gupta. 2016. Launch and iterate: Reducing prediction churn. In *Advances in Neural Information Processing Systems*, pages 3179–3187.

Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H Chi, and Alex Beutel. 2018. Counterfactual fairness in text classification through robustness. *arXiv preprint arXiv:1809.10610*.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10):2451–2471.

Alex Graves. 2012. *Supervised sequence labelling with recurrent neural networks*, volume 385. Springer.

Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In *Proc. of NeurIPS*, pages 3315–3323.

Johannes Hellrich and Udo Hahn. 2016. Bad company—neighborhoods in neural embedding spaces considered harmful. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2785–2796.

Johannes Hellrich, Bernd Kampe, and Udo Hahn. 2019. The influence of down-sampling strategies on svd word embedding stability. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*, pages 18–26.

Young-Bum Kim, Karl Stratos, and Dongchan Kim. 2017. Adversarial adaptation of synthetic or stale data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1297–1307.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proc. of EMNLP*, pages 1746–1751.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270.

Megan Leszczynski, Avner May, Jian Zhang, Sen Wu, Christopher Aberger, and Christopher Re. 2020. Understanding the downstream instability of word embeddings. In *Proceedings of Machine Learning and Systems 2020*, pages 262–290.

Avner May, Jian Zhang, Tri Dao, and Christopher Ré. 2019. On the downstream performance of compressed word embeddings. In *Advances in neural information processing systems*, pages 11805–11816.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Margaret Mitchell, Simone Wu, Andrew Zaldivar, Parker Barnes, Lucy Vasserman, Ben Hutchinson, Elena Spitzer, Inioluwa Deborah Raji, and Timnit Gebru. 2019. Model cards for model reporting. In *Proc. of FATML*, pages 220–229.

SPFGH Moen and Tapio Salakoski2 Sophia Ananiadou. 2013. Distributional semantics resources for biomedical text processing. *Proceedings of LBM*, pages 39–44.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proc. of EMNLP*, pages 2799–2804.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*, pages 1532–1543.

Bénédicte Pierrejean and Ludovic Tanguy. 2018. Predicting word embeddings variability. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 154–159.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535.

Alan Ramponi and Barbara Plank. 2020. Neural unsupervised domain adaptation in nlp—a survey. *arXiv*, pages arXiv–2006.

Anthony Rios. 2020. Fuzze: Fuzzy fairness evaluation of offensive language classifiers on african-american english. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 881–889.

Anna Rogers, Shashwath Hosur Ananthakrishna, and Anna Rumshisky. 2018. What's in your embedding, and how it predicts task performance. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2690–2703.

Morgan Klaus Scheuerman, Jacob M. Paul, and Jed R. Brubaker. 2019. How computers see gender: An evaluation of gender classification in commercial facial analysis services. *Proc. ACM Hum.-Comput. Interact.*, 3(CSCW), November.

Roy Schwartz, Jesse Dodge, Noah A Smith, and Oren Etzioni. 2019. Green ai. *arXiv preprint arXiv:1907.10597*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93.

Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the type and target of offensive posts in social media. *arXiv preprint arXiv:1902.09666*.

Indre Zliobaite. 2015. A survey on measuring indirect discrimination in machine learning. *arXiv preprint arXiv:1511.00148*.