Using Feedforward and Recurrent Neural Networks to Predict a Blogger's Age

Tim Moon
Stanford University
tym1@stanford.edu

Eric Liu
Stanford University
ericql@stanford.edu

Abstract

Predicting the age of a blogger based on the text of their writing is a difficult task due to the fluidity of age identity and the effect of aging on writing styles. We propose feedforward and recurrent neural network frameworks to address this problem without enforcing human-generated features and find that shallow networks suffice for this problem. Results suggest that a scaled bag-of-words feedforward neural network model is better suited for age prediction than a pre-processed stacked long short-term memory model, possibly because word choice may be more indicative of author age than the meaning of the text.

1 Introduction

Given a blog post, we seek to predict demographic characteristics of the author, namely the author's age. This can be a fraught endeavor since age identity is a fluid social variable that does not always correspond to chronological or biological age [7]. Sociological concerns aside, however, there is a substantial body of work that attempts to predict age using writing style and content. Previous experiments have used human guesses [7], a winnow algorithm with a curated dictionary [12], naive Bayes with slang usage [3], support vector machines with word and character n-grams [8, 10], logistic regression with lexical features and blogger behavior [11], and ridge regression with unigrams [6]. These studies have been performed on data sets from Blogger, Twitter, LiveJournal, and Netlog and they show that older writers tend to use less slang, write longer sentences, include more hyperlinks, and use more inclusive pronouns ("we" instead of "I"). However, these methods become less effective at identifying bloggers over the age of 30, suggesting that a person's writing style becomes more static around that age. We are not aware of any published attempts to approach this problem with deep learning methods, so we attempt to apply the flexibility and predictive power of neural networks to fill this gap in the sociolinguistic research. Neural networks have the additional advantage that they can function without human-generated features and with a smaller feature space than n-grams.

2 Problem Statement

We attempt to predict the age of a blogger given the text of a blog post. Data is drawn from the Blog Authorship Corpus, a collection of blog posts from Blogger [12]. This data set spans a period from 1999 to 2003 and consists of 681,288 posts from 19,320 blogs, for a total of over 140 million words. Each blogger is tagged with gender, age, and astrological sign (which can be used to establish birth month) and each post is tagged with the post date. 65% of the bloggers are also tagged with their profession. There are an equal number of male and female bloggers and each of the bloggers belong to one of three discrete age groups: 43% are between the ages of 13 and 17, 42% between 23 and 27, and 15% between 33 and 47. We seek to solve a classification problem to predict a blogger's age group and a regression problem to predict their numerical age. Method effectiveness will be evaluated using the macro-average F-score for classification and the mean absolute error for regression.

3 Data Processing

After removing improperly formatted blog posts in preliminary processing, we are left with a data set of 659,247 blog posts. The number of blog posts per blogger follows a power-law distribution:

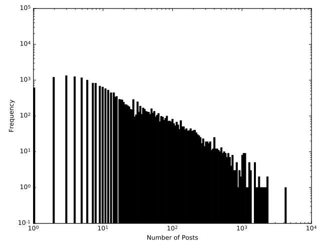

Figure 1: Histogram of number of posts per blogger, taken over the entire data set.

To avoid imbalances arising from this power-law behavior, we treat each blog post as an independent data sample. In order to validate our models, we divide the data set into training (70.0%), validation (19.3%), and test (10.7%) sets, split by blogger to prevent data leakage. As expected, the distribution of blogger ages exhibits discrete 10-20, 20-30, and 30-50 age groups:

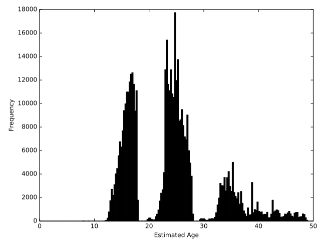

Figure 2: Histogram of age at time of blog post, taken over the training set.

In order to make the blog posts more amenable to analysis, we tokenize them by word and puncutation mark. These tokens are case insensitive and limited to the ASCII character set. URLs are denoted with `urllink` and numerals with `N`.[1] The five most common tokens are the period, the comma, `i`, `the`, and `to`. Token frequencies obey a power-law distribution:

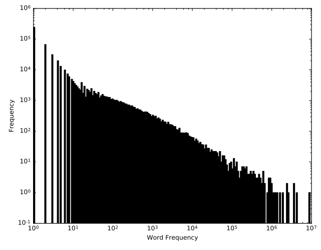

Figure 3: Histogram of token frequency, taken over the training set.

To keep the number of tokens manageable, the 10,000 most common tokens were retained and the remainder converted to the unknown token `UNK`.

[1]Single-digit numbers are denoted with `N`, two-digit numbers with `NN`, and so on.

4 Models

4.1 Notation

Our model recognizes $V = 10,001$ unique tokens and each blog post falls into one of $n = 3$ age groups. A post is represented with a list of ℓ tokens $\{w_1, \cdots, w_\ell\}$, the blogger's age group with a one-hot vector $\mathbf{y} \in \mathbb{R}^n$, and their age at the time of the post with $a > 0$. For the classification problem we seek a probability vector $\hat{\mathbf{y}} \in \mathbb{R}^n$ that approximates $\mathbf{y}$ and for the regression problem we seek $\hat{a} \in \mathbb{R}$ that approximates a. We will use the row vector convention in this report.

4.2 Scaled Bag-of-Words Feedforward Neural Network

Given a post, we begin by computing a real-valued vector $\mathbf{x} \in \mathbb{R}^d$ with the linear transformation

$$\mathbf{x} = \mathbf{v}\mathbf{L} \tag{1}$$

The embedding matrix $\mathbf{L} \in \mathbb{R}^{V \times d}$ is initialized for $d = 300$ with GloVe word vectors trained on the Common Crawl data set [9]. If a token does not correspond to any pre-trained word vector, a random word vector is generated with Xavier initialization [2]. The unembedded vector $\mathbf{v} \in \mathbb{R}^V$ is constructed with a scaled bag-of-words approach, namely by counting the number of times each token appears and scaling by a post length cost $c_{\text{pl}}(\ell)$,

$$\mathbf{v} = c_{\text{pl}}(\ell) \sum_{t=1}^{\ell} \mathbf{e}_{w_t} \tag{2}$$

Note that if $c_{\text{pl}}(\ell) = 1$, then $\mathbf{x}$ is simply the sum of the post's word vectors. Alternatively, if $c_{\text{pl}}(\ell) = 1/\ell$, then $\mathbf{x}$ is the mean of the word vectors. The former vector is highly sensitive to post length and the latter is independent of it. We hypothesize that word selection and post length are both important factors to consider in a model, so we consider length costs with intermediate asymptotic behavior.

Once the post embedding vector is constructed, it is inputted into a feedforward neural network with L hidden layers of size H. The ith hidden layer has weight matrix $\mathbf{W}^{(i)} \in \mathbb{R}^{H \times H}$ and bias vector $\mathbf{b}^{(i)} \in \mathbb{R}^H$.[2] Denoting $\mathbf{h}^{(0)} = \mathbf{x}$, the ith hidden state is given by

$$\mathbf{h}^{(i)} = \tanh\left(\mathbf{h}^{(i-1)}\mathbf{W}^{(i)} + \mathbf{b}^{(i)}\right) \tag{3}$$

For classification, we apply a softmax output layer with weight matrix $\mathbf{U} \in \mathbb{R}^{H \times n}$ and bias vector $\mathbf{c} \in \mathbb{R}^n$,

$$\hat{\mathbf{y}} = \text{softmax}\left(\mathbf{h}^{(L)}\mathbf{U} + \mathbf{c}\right) \tag{4}$$

Parameters for this model are evaluated with the cross entropy loss,

$$J_{\text{CE}}(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{j=1}^{n} \mathbf{y}_j \log \hat{\mathbf{y}}_j \tag{5}$$

For regression, we apply a dot product output layer with weight vector $\mathbf{u} \in \mathbb{R}^H$ and bias $c \in \mathbb{R}$,

$$\hat{a} = \mathbf{h}^{(L)}\mathbf{u}^T + c \tag{6}$$

Parameters are evaluated with the L2 loss,

$$J_{\text{L2}}(\hat{a}, a) = (\hat{a} - a)^2 \tag{7}$$

For the sake of brevity, we shall refer to the scaled bag-of-words feedforward neural networks as the "FNN classification" and "FNN regression" models.

4.3 Pre-processed Stacked Long Short-Term Memory Neural Network

One limitation of the FNN models is that a bag-of-words approach overlooks word ordering. We propose a recurrent neural network model with L feedforward hidden layers and M long short-term memory (LSTM) hidden layers [4], each of which have size H. To simplify calculation, we only consider the first T tokens of each post.[3] The input vector for word t in post p is constructed with

[2]The first weight matrix $\mathbf{W}^{(1)}$ has dimension $d \times H$.

[3]If a token has fewer than T tokens, the beginning is padded with `UNK` tokens.

$\mathbf{x}_t = \mathbf{e}_{w_t}\mathbf{L}$. Denoting $\mathbf{h}_t^{(0)} = \mathbf{x}_t$, the ith feedforward hidden state is given by

$$\mathbf{h}_t^{(i)} = \tanh\left(\mathbf{h}_t^{(i-1)}\mathbf{W}^{(i)} + \mathbf{b}^{(i)}\right) \tag{8}$$

This layer has weight matrix $\mathbf{W}^{(i)} \in \mathbb{R}^{H\times H}$ and bias vector $\mathbf{b}^{(i)} \in \mathbb{R}^{H}$.[4] Denoting $LSTM_t^{(0)} = \mathbf{h}_t^{(L)}$ and $LSTM_0^{(m)} = 0$, the mth LSTM hidden state is given by

$$\mathbf{i}_t^{(m)} = \sigma\left(LSTM_t^{(m-1)}\mathbf{W}_i^{(m)} + LSTM_{t-1}^{(m)}\mathbf{U}_i^{(m)}\right) \tag{9}$$

$$\mathbf{f}_t^{(m)} = \sigma\left(LSTM_t^{(m-1)}\mathbf{W}_f^{(m)} + LSTM_{t-1}^{(m)}\mathbf{U}_f^{(m)}\right) \tag{10}$$

$$\mathbf{o}_t^{(m)} = \sigma\left(LSTM_t^{(m-1)}\mathbf{W}_o^{(m)} + LSTM_{t-1}^{(m)}\mathbf{U}_o^{(m)}\right) \tag{11}$$

$$\tilde{\mathbf{c}}_t^{(m)} = \tanh\left(LSTM_t^{(m-1)}\mathbf{W}_c^{(m)} + LSTM_{t-1}^{(m)}\mathbf{U}_c^{(m)}\right) \tag{12}$$

$$\mathbf{c}_t^{(m)} = \mathbf{f}_t^{(m)} \circ \mathbf{c}_{t-1}^{(m)} + \mathbf{i}_t^{(m)} \circ \tilde{\mathbf{c}}_t^{(m)} \tag{13}$$

$$LSTM_t^{(m)} = \mathbf{o}_t^{(m)} \circ \tanh\left(\mathbf{c}_t^{(m)}\right) \tag{14}$$

The LSTM cell has weight matrices $\mathbf{W}_i^{(m)}, \mathbf{U}_i^{(m)}, \mathbf{W}_f^{(m)}, \mathbf{U}_f^{(m)}, \mathbf{W}_o^{(m)}, \mathbf{U}_o^{(m)}, \mathbf{W}_c^{(m)}, \mathbf{U}_c^{(m)} \in \mathbb{R}^{H\times H}$. For classification, we apply a softmax output layer with weights $\mathbf{V} \in \mathbb{R}^{H\times n}$ and bias $\mathbf{d} \in \mathbb{R}^n$,

$$\hat{\mathbf{y}}_t = \text{softmax}\left(LSTM_t^{(M)}\mathbf{V} + \mathbf{d}\right) \tag{15}$$

The parameters are evaluated with the weighted cross entropy loss,

$$J_{\text{WCE}} = \sum_{t=1}^{T} t\, J_{\text{CE}}(\hat{\mathbf{y}}_t, \mathbf{y}) \tag{16}$$

The model is summarized in the following diagram:

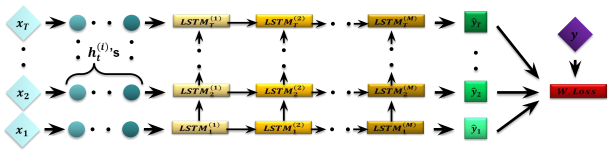

Figure 4: Cartoon of pre-processed stacked LSTM model.

This model (which we call the "LSTM model" for the sake of brevity) makes two alterations to the standard stacked LSTM model. First, instead of using the word vectors directly as inputs, we pre-process them with several feedforward hidden layers. Second, instead of using the final prediction exclusively, we use a weighted average of losses from all time step predictions, giving more weight to later time steps. The key motivation is that although LSTM models theoretically incorporate all relevant information from previous time steps, early information is often lost in practice. Thus, using a weighted average ensures that the model does not ignore early words while still giving the most weight to later predictions.

4.4 Implementation

The FNN and LSTM models are implemented with TensorFlow [1]. Weights are initialized with Xavier initialization [2] and biases are initialized to zero. The model parameters – the weights, biases, and embedding matrix – are trained by minimizing batched loss functions, i.e. the average loss function with several data samples, using Adam [5]. To prevent overfitting, we apply L2 regularization on the weight matrices and dropout on the input and hidden states [13]. In addition, we only use the 20-30 age group for the regression problem to reduce adverse effects from the non-uniform age distribution.

[4]The first weight matrix $\mathbf{W}^{(i)}$ has dimensions $d \times H$.

5 Results

5.1 Post Length Cost for FNN Models

In order to investigate the effect of different post costs we experimented on the FNN classification model with one hidden layer of size 100, batch size 64, Adam learning rate 10^{-3}, dropout ratio 0.9, and L2 regularization parameter 10^{-3}:

$c_{\text{pl}}(\ell)$	Train Accuracy	Validation Accuracy
1	68.0%	64.1%
$1/\ell$	68.5%	64.9%
$1/\sqrt{\ell}$	70.6%	65.2%
$1/\log(\ell+1)$	70.9%	65.3%

The best results were obtained when the post length cost was asymptotically between 1 and $1/\ell$, supporting our hypothesis that both the word selection and post length provide useful information. In all subsequent experiments, we used the post length cost $c_{\text{pl}}(\ell) = 1/\log(\ell+1)$.

5.2 Hyperparameter Search for FNN Classification Model

A hyperparameter grid search was performed for the FNN classification model by varying the number of hidden layers and the hidden layer size:

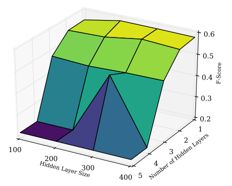

Figure 5: Validation macro-average F-score from a hyperparameter grid search for the FNN classification model.

These trials were performed with batch size 256, Adam learning rate 10^{-2}, dropout ratio 0.9, and L2 regularization parameter 2×10^{-2}. We see that shallower neural networks outperformed deeper ones, likely because networks with a smaller parameter space could converge more quickly to an approximately optimal solution. Since each network was given roughly the same amount of training time, it is reasonable that those with fewer hidden layers produced better results. In addition, observe that the best hidden layer size is around 300, which coincides with the embedding size of 300. This suggests that setting the hidden layer size equal to the embedding size allows the network to preserve information from the post embeddings without creating superfluous parameters. A hyperparameter random search was performed by varying the number of hidden layers, the hidden layer size, the dropout ratio, the Adam learning rate, and the L2 regularization parameter (the batch size was maintained at 256). The best model obtained had the following parameters:

Hyperparameter	Value
Number of hidden layers	1
Hidden layer size	288
Dropout ratio	0.97
Adam learning rate	1.13×10^{-4}
L2 regularization parameter	1.16×10^{-3}

Note that number of hidden layers and the hidden layer size are consistent with our hyperparameter grid search results. When this model trained with 24 epochs of the training set, it reported the following learning curves:

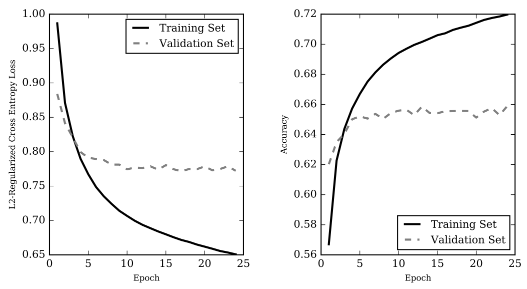

Figure 6: Training curves for best observed FNN classification model.

This model had 71.0% training accuracy , 65.6% validation accuracy, and 64.7% test accuracy. The test set confusion matrix was:

	10-20	20-30	30-50
10-20	17393	6690	455
20-30	5709	24861	3068
30-50	939	8049	3466

The test set precision and recall of each age group are summarized below:

Age Group	Precision	Recall
10-20	72.3%	70.1%
20-30	62.8%	73.9%
30-50	49.6%	27.8%

This model had a test set macro-average F-score of 0.595. It achieved good performance with the 10-20 and 20-30 age groups, but struggled distinguishing between the 20-30 and 30-50 age groups.

5.3 Perturbation Test for FNN Classification Model

Since previous studies involving our formulation of the age prediction problem do not report results with a holdout test set or cross validation [3, 12],[5] we performed perturbation tests to evaluate the significance of our best model. We ran 50 trials of training the FNN model with the same training set, but with randomly permuted age groups. The mean validation accuracy was 40.9% with a standard deviation of 7.3%. The 65.6% accuracy attained by our model thus has a test statistic of 3.39 and a p-value of 0.0003, allowing us to reject the null hypothesis at the 0.1% significance level.

5.4 FNN Regression Model

The hyperparameter grid and random searches discussed in Subsection 5.2 were repeated for the FNN regresion model. Results from the grid search are presented below:

[5]Other experiments with the age classification problem use different age groups.

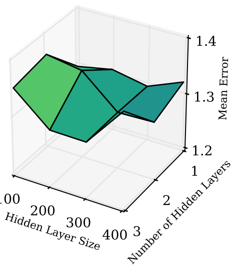

Figure 7: Validation macro-average F-score from a hyperparameter grid search for the FNN regression model.

The grid search results appear to show no structure. The best model observed in the random search had the following parameters:

Hyperparameter	Value
Number of hidden layers	1
Hidden layer size	115
Dropout ratio	0.92
Adam learning rate	2×10^{-4}
L2 regularization parameter	8×10^{-3}

This model has a mean absolute error of 1.28 on the training set, 1.19 on the validation set, and 1.30 on the test set. The fact that the error on the validation set is much smaller than on the training and test tests suggests that this choice of parameters is an artifact of random parameter initialization rather than a statistically significant result. In fact, inspection of the predictions shows that the model tends to simply predict ages close to 25, which is the mean age for the 20-30 age group. We thus conclude that this model is not effective for this problem.

5.5 LSTM Model

The LSTM model was trained with one feedforward hidden layer, one LSTM hidden layer, hidden layer size 250, dropout ratio 0.9, batch size 256, Adam learning rate 10^{-4}, and L2 regularization parameter 10^{-3}. We obtain the following learning curves:

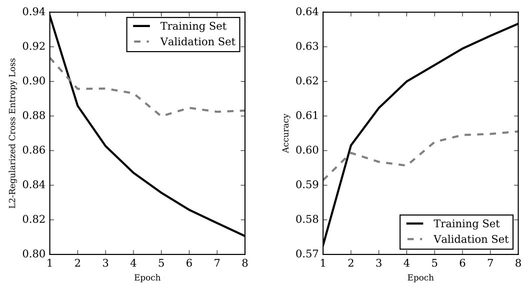

Figure 8: Training curves for pre-processed stacked LSTM model.

This model had 62.9% training accuracy, 60.5% validation accuracy, and 65% test accuracy. The validation set confusion matrix is:

	10-20	20-30	30-50
10-20	22369	20351	237
20-30	7656	53035	1101
30-50	1314	19558	1355

The validation set precision and recall of each age group are summarized below:

Age Group	Precision	Recall
10-20	71.4%	52.1%
20-30	57.1%	85.8%
30-50	30.3%	6.1%

This model reported a macro-average F-score of 0.532. The pre-processed stacked LSTM model had similar behavior as the FNN classification model, i.e. it struggled with distinguishing between the 20-30 and 30-50 age groups. However, it exhibited a stronger prediction bias toward the 20-30 age group. This result is worse than with the FNN classification model, although we note that this is not a fair comparison since we did not perform a thorough hyperparameter search. Nonetheless, it is suggestive of the difference between the FNN and LSTM approaches. The FNN bag-of-word approach distinguishes the exact vocabulary set in blog posts whereas the LSTM model attempts to extract the meaning from a post. We postulate that the choice of words is more important for the age prediction problem than the meaning. For instance, a 13-year-old and 47-year-old could both write about health and lifestyle, but their writing styles and word choice are likely to differ. If this is the case, we would expect the FNN model to outperform the LSTM model.

6 Conclusions

We conclude that the FNN classification model is an effective approach for the age prediction problem. In particular, we find that post length scaling is an effective method to capture both the word choice and length of a post and that shallow neural networks tend to outperform deeper ones. The effectiveness of the algorithm is statistically significant to the 0.1% significance level. We also report that the FNN regression model fails for this problem and that the LSTM model is functional, albeit not as effective as the FNN classification model. We propose three extensions to our work that may yield fruitful results. First, due to the poor performance with the recognizing the 30-50 age group, it is reasonable to isolate this problem and separately train a binary classifier tuned specifically for this age group. Second, employing a mixed modeling approach that includes both curated features and word vector inputs may improve both performance and interpretability. Lastly, model strength could potentially be improved by exploring time-dependent models that capture the variability among each age group's adoption rate of new words (e.g. slang) and cultural trends.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[3] Sumit Goswami, Sudeshna Sarkar, and Mayur Rustagi. Stylometric analysis of bloggers age and gender. In *Proceedings of the Third International AAAI Conference on Weblogs and Social Media*, 2009.

[4] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[5] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[6] Dong Nguyen, Rilana Gravel, Dolf Trieschnigg, and Theo Meder. TweetGenie: automatic age prediction from tweets. *ACM SIGWEB Newsletter*, 4(4), 2013.

[7] Dong-Phuong Nguyen, RB Trieschnigg, AS Doğruöz, Rilana Gravel, Mariët Theune, Theo Meder, and FMG de Jong. Why gender and age prediction from tweets is hard: Lessons from a crowdsourcing experiment. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Association for Computational Linguistics, 2014.

[8] Claudia Peersman, Walter Daelemans, and Leona Van Vaerenbergh. Predicting age and gender in online social networks. In *Proceedings of the 3rd International Workshop on Search and Mining User-generated Contents*, pages 37–44. ACM, 2011.

[9] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing*, pages 1532–1543, 2014.

[10] Delip Rao, David Yarowsky, Abhishek Shreevats, and Manaswi Gupta. Classifying latent user attributes in Twitter. In *Proceedings of the 2nd International Workshop on Search and Mining User-generated Contents*, pages 37–44. ACM, 2010.

[11] Sara Rosenthal and Kathleen McKeown. Age prediction in blogs: A study of style, content, and online behavior in pre-and post-social media generations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 763–772. Association for Computational Linguistics, 2011.

[12] Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, volume 6, pages 199–205, 2006.

[13] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.