Gender Classification from Text

Michael Tran, Matthew Lai

Abstract—The problem of automatically classifying the gender of a blog author has important applications in many commercial domains. Existing systems mainly use features such as words, word classes, and POS n-grams for classification learning. We leverage the solution presented in the paper "Improving Gender Classification of Blog Authors" by A. Mukherjee et al. [1] and implement POS Sequence Pattern Mining and Ensemble Feature Selection. We also propose a novel method named "Supervised Semi-Supervised" (SSS) learning that improves the classification accuracy.

Index Terms—Gender, Gender Classification, Supervised Semi-Supervised Learning, POS Sequence Pattern Mining, Ensemble Feature Selection, Feature Engineering, Feature Mining

1 Introduction

Weblogs, commonly known as blogs, contain informal writings of personal diaries. In the recent world, many companies leverage information from blogs to gain market intelligence which can be exploited in targeted advertising and product development. Knowing the gender of a blog author specifically, allows companies to know what products and services are disliked by men and women. It can also help companies find what topics or products that are most talked about by a gender. This information could be crucial to help companies improve their products and in turn make more profit.

Making gender classification from blogs is harder than making gender classification from essays, news, or papers. This is due to the fact that people don't usually use formal sentences, correct grammar, abbreviations, and correct spelling when they are writing blogs. Blogs will also contain slang words and phrases which are not typically seen in other texts. In general, blogs are also not as long and constructed as essays. Blogs are typically short, and unstructured, and consist of mostly informal sentences. Due to these reasons, we can see why gender classification of blog posts is such a harder problem than gender classification of traditional formal text.

A. Halevy et al. [2] argues that introducing more data for some tasks could prove to have a positive effect on classifiers. Adding additional data can increase the accuracy to a certain point, but also have diminishing values or detrimental values at a certain point. We propose a novel method named "Supervised Semi-Supervised" learning where we attempt to leverage additional blog data to increase the classification accuracy without overfitting.

Our experimental results are based on two different blog datasets. For this paper, we name them Dataset 1 and Dataset 2.

- Dataset 1: Real life blog data set collected from a large number of blog hosting sites. This dataset was used in Mukherjee et al. [1] [8]
- Dataset 2: Blog Authorship Corpus – consists of the collected posts of 19,320 bloggers gathered from blogger.com in August 2004. The corpus incorporates a total of 681,288 posts and over 140 million words – or approximately 35 posts and 7250 words per person. [9] In our experiments, we extract the first 17443 blog posts from this dataset.

In this paper, Dataset 1 is considered our "base" dataset and Dataset 2 is our "additional" dataset that we will add-on to Dataset 1.

2 Related Work

There have been several papers on gender classification of blogs. [3] [4] [5] [6] They used features like function/content words, POS tag features, word classes, content word classes, results of dictionary-based content analysis, POS unigrams, and personality types to capture stylistic behavior of authors' writings for classifying gender.

However, these works use only one or a subset of these classes of features. A. Mukherjee et Al. [1] proposes a solution to use all of these features. They propose a POS sequence pattern mining algorithm and an ensemble feature selection algorithm that improves gender classification accuracy from previous methods significantly.

3 Feature Engineering and Mining

Before continuing, we note that the blog posts in the data sets were seen to have inconsistent punctuactions. So as a preprocessing step, we apply punctuation normalization to ensure that punctuation remains consistent throughout all blog samples. For example, we change the Unicode double quote [“] to the UTF-8 double quote ["] and we change the Unicode single quote [‘] to the UTF-8 single quote ['].

3.1 F-measure

F-measure is a unitary measure used to measure a text's relative contextuality (implicitness), as opposed to its formality (explicitness). Contextuality and formality can be captured by certain parts of speech. A lower score of F-measure indicates contextuality, which means a greater relative use of pronouns, verbs, adverbs, and interjec-

tions. A higher F-measure indicates formality, which means a greater relative use of nouns, adjectives, prepositions, and articles. F-measure is defined by the following formula:

$$F = 0.5 * [(freq.noun + freq.adj + freq.prep + freq.art) - (freq.pron + freq.verb + freq.adv + freq.int) + 100]$$

We use F-measure as one of our features. Males commonly get a higher score because they prefer a more formal style when writing blogs, whereas females get a lower score because they prefer a more contextual style when writing blogs.

3.2 Stylistic Features

These are features that capture people's writing styles. The style of writing is typically captured by three types of features: part of speech, words, and blog words. POS Sequence Pattern Mining covers part-of-speech so we focus on words and blog words for this section.

We extract two different types of tokenized text from the same blog text.

1. Tokenized Text 1: stopwords are removed from the text and then the remaining text is lemmatized. (Grouping together the inflected forms of a word.) The resultant words are then vectorized using 1-grams.
2. Tokenized Text 2: the text is simply tokenized using a widely known NLP tool called NLTK. [7] The resultant words are then vectorized using 2, 3-grams.

We also extract the word counts of each blog post and use it as a feature.

3.3 Gender Preferential Features

These features consist of a set of signals from language that have been shown to belong more to either females or males. Women tend to use signals that make more frequent use of emotionally intensive adverbs and adjectives like "so", "terribly", "awfully", "dreadfully". On the other hand, men's conversational patterns express "independence". In brief, the language expressed by men is more proactive while the language used by females is more reactive. We use GPF as features and Table 1 shows the feature assignments.

f1	Words ending with *able*
f2	Words ending with *al*
f3	Words ending with *ful*
f4	Words ending with *ible*
f5	Words ending with *ic*
f6	Words ending with *ive*
f7	Words ending with *less*
f8	Words ending with *ly*
f9	Words ending with *ous*
f10	*Sorry* words such as sorry, penitent, contrite, repentant, remorseful, regretful, etc

Table 1 – Gender Preferential Feature assignment.

3.4 Factor Analysis

Word factor analysis refers to the process of finding groups of similar words that tend to occur in similar documents. Table 2 shows each factor and the words associated with the factor.

Factor	Words
Conversation	'know', 'people', 'think', 'person', 'tell', 'feel', 'friends', 'talk', 'new', 'talking', 'mean', 'ask', 'understand', 'feelings', 'care', 'thinking', 'friend', 'relationship', 'realize', 'question', 'answer', 'saying'
Home	'woke', 'home', 'sleep', 'today', 'eat', 'tired', 'wake', 'watch', 'watched', 'dinner', 'ate', 'bed', 'day', 'house', 'tv', 'early', 'boring', 'yesterday', 'watching', 'sit'
Family	'years', 'family', 'mother', 'children', 'father', 'kids', 'parents', 'old', 'year', 'child', 'son', 'married', 'sister', 'dad', 'brother', 'moved', 'age', 'young', 'months', 'three', 'wife', 'living', 'college', 'four', 'high', 'five', 'died', 'six', 'baby', 'boy', 'spend', 'christmas'
Time	'friday', 'saturday', 'weekend', 'week', 'sunday', 'night', 'monday', 'tuesday', 'thursday', 'wednesday', 'morning', 'tomorrow', 'tonight', 'evening', 'days', 'afternoon', 'weeks', 'hours', 'july', 'busy', 'meeting', 'hour', 'month', 'june'
Work	'work', 'working', 'job', 'trying', 'right', 'met', 'figure', 'meet', 'start', 'better', 'starting', 'try', 'worked', 'idea'
Past Actions	'said', 'asked', 'told', 'looked', 'walked', 'called', 'talked', 'wanted', 'kept', 'took', 'sat', 'gave', 'knew', 'felt', 'turned', 'stopped', 'saw', 'ran', 'tried', 'picked', 'left', 'ended'
Games	'game', 'games', 'team', 'win', 'play', 'played', 'playing', 'won', 'season', 'beat', 'final', 'two', 'hit', 'first', 'video', 'second', 'run', 'star', 'third', 'shot', 'table', 'round', 'ten', 'chance', 'club', 'big', 'straight'
Internet	'site', 'email', 'page', 'please', 'website', 'web', 'post', 'link', 'check', 'blog', 'mail', 'information', 'free', 'send', 'comments', 'comment', 'using', 'internet', 'online', 'name', 'service', 'list', 'computer', 'add', 'thanks', 'update', 'message'
Location	'street', 'place', 'town', 'road', 'city', 'walking', 'trip', 'headed', 'front', 'car', 'beer', 'apartment', 'bus', 'area', 'park', 'building', 'walk', 'small', 'places', 'ride', 'driving', 'looking', 'local', 'sitting', 'drive', 'bar', 'bad', 'standing', 'floor', 'weather', 'beach', 'view'
Fun	'fun', 'im', 'cool', 'mom', 'summer', 'awesome', 'lol', 'stuff', 'pretty', 'ill', 'mad', 'funny', 'weird'
Food/Clothes	'food', 'eating', 'weight', 'lunch', 'water', 'hair', 'life', 'white', 'wearing', 'color', 'ice', 'red', 'fat', 'body', 'black', 'clothes', 'hot', 'drink', 'wear', 'blue', 'minutes', 'shirt', 'green', 'coffee', 'total', 'store', 'shopping'
Poetic	'eyes', 'heart', 'soul', 'pain', 'light', 'deep', 'smile', 'dreams', 'dark', 'hold', 'hands', 'head', 'hand', 'alone', 'sun', 'dream', 'mind', 'cold', 'fall', 'air', 'voice', 'touch', 'blood', 'feet', 'words', 'hear', 'rain', 'mouth'
Books / Movies	'book', 'read', 'reading', 'books', 'story', 'writing', 'written', 'movie', 'stories', 'movies', 'film', 'write', 'character', 'fact', 'thoughts', 'title', 'short', 'take', 'wrote'
Religion	'god', 'jesus', 'lord', 'church', 'earth', 'world', 'word', 'lives', 'power', 'human', 'believe', 'given', 'truth', 'thank', 'death', 'evil', 'own', 'peace', 'speak', 'bring', 'truly'
Romance	'forget', 'forever', 'remember', 'gone', 'true', 'face', 'spent', 'times', 'love', 'cry', 'hurt', 'wish', 'loved'
Swearing	'shit', 'fuck', 'fucking', 'ass', 'bitch', 'damn', 'hell', 'sucks', 'stupid', 'hate', 'drunk', 'crap', 'kill', 'guy', 'gay', 'kid', 'sex', 'crazy', 'cunt', 'nigger', 'nigga', 'asshole', 'pussy', 'dick', 'dickhead', 'faggot', 'fag'
Politics	'bush', 'president', 'iraq', 'kerry', 'war', 'american', 'political', 'states', 'america', 'country', 'government', 'john', 'national', 'news', 'state', 'support', 'issues', 'article', 'michael', 'bill', 'report', 'public', 'issue', 'history', 'party', 'york', 'law', 'major', 'act', 'fight', 'poor'
Music	'music', 'songs', 'song', 'band', 'cd', 'rock', 'listening', 'listen', 'show', 'favorite', 'radio', 'sound', 'heard', 'shows', 'sounds', 'amazing', 'dance'
School	'school', 'teacher', 'class', 'study', 'test', 'finish', 'english', 'students', 'period',

	'paper', 'pass'
Business	'system', 'based', 'process', 'business', 'control', 'example', 'personal', 'experience', 'general'
Positive	'absolutely', 'abundance', 'ace', 'active', 'admirable', 'adore', 'agree', 'amazing', 'appealing', 'attraction', 'bargain', 'beaming', 'beautiful', 'best', 'better', 'boost', 'breakthrough', 'breeze', 'brilliant', 'brimming', 'charming', 'clean', 'clear', 'colorful', 'compliment', 'confidence', 'cool', 'courteous', 'cuddly', 'dazzling', 'delicious', 'delightful', 'dynamic', 'easy', 'ecstatic', 'efficient', 'enhance', 'enjoy', 'enormous', 'excellent', 'exotic', 'expert', 'exquisite', 'flair', 'free', 'generous', 'genius', 'great', 'graceful', 'heavenly', 'ideal', 'immaculate', 'impressive', 'incredible', 'inspire', 'luxurious', 'outstanding', 'royal', 'speed', 'splendid', 'spectacular', 'superb', 'sweet', 'sure', 'supreme', 'terrific', 'treat', 'treasure', 'ultra', 'unbeatable', 'ultimate', 'unique', 'wow', 'zest'
Negative	'wrong', 'stupid', 'bad', 'evil', 'dumb', 'foolish', 'grotesque', 'harm', 'fear', 'horrible', 'idiot', 'lame', 'mean', 'poor', 'heinous', 'hideous', 'deficient', 'petty', 'awful', 'hopeless', 'fool', 'risk', 'immoral', 'risky', 'spoil', 'spoiled', 'malign', 'vicious', 'wicked', 'fright', 'ugly', 'atrocious', 'moron', 'hate', 'spiteful', 'meager', 'malicious', 'lacking'
Emotion	'aggressive', 'alienated', 'angry', 'annoyed', 'anxious', 'careful', 'cautious', 'confused', 'curious', 'depressed', 'determined', 'disappointed', 'discouraged', 'disgusted', 'ecstatic', 'embarrassed', 'enthusiastic', 'envious', 'excited', 'exhausted', 'frightened', 'frustrated', 'guilty', 'happy', 'helpless', 'hopeful', 'hostile', 'humiliated', 'hurt', 'hysterical', 'innocent', 'interested', 'jealous', 'lonely', 'mischievous', 'miserable', 'optimistic', 'paranoid', 'peaceful', 'proud', 'puzzled', 'regretful', 'relieved', 'sad', 'satisfied', 'shocked', 'shy', 'sorry', 'surprised', 'suspicious', 'thoughtful', 'undecided', 'withdrawn'

Table 2 – Factor Analysis Feature assignment.

3.5 POS Sequence Pattern Mining

The part-of-speech sequence pattern features is a sequence of consecutive POS tags that satisfy the constraints: minimum support and minimum adherence. They are related to POS n-grams, but are not the same. POS sequence patterns not only include POS n-grams, but also captures sequence regularities of the POS n-grams. The main idea of the algorithm is to perform a level-wise search for such patterns, which are POS sequences with minimum support and minimum adherence.

The support of a pattern is simply the proportion of documents that contain the pattern. If a pattern appears too few times, it is probably spurious. A pattern is called a frequent sequence if it satisfies the minimum support. The adherence of a pattern is measured using the symmetrical conditional probability (SCP). We use the formula for fairSCP in our implementation. [1]

$$fairSCP(x_1, \dots, x_n) = \frac{P(x_1, \dots, x_n)^2}{\frac{1}{n-1}\sum_{i=1}^{n-1} P(x_1, \dots, x_i)P(x_{i+1}, \dots, x_n)}$$

FairSCP measures the adherence strength of POS tags in a sequence. If the fairSCP value is high, it means that the sequence is more dominant.

We implemented this POS sequence pattern mining algorithm and use the same values as Mukherjee et al. [1] where minimum support = 30% and minimum adherence = 20%. The algorithm was run against Dataset 1 and we found 194 POS sequence patterns.

For POS tagging, we used the state-of-the-art NLP library called flair. [14] Flair is a very powerful NLP library that improved our classification results considerably when compared to NLTK. However, even though it is much more accurate, it does take a considerably longer time to process the text corpus.

3.6 Ensemble Feature Selection

EFS uses a number of feature selection criteria to rank the features following the filter model. Upon ranking, the algorithm generates some candidate feature subsets which are used to find the final feature set based on the classification accuracy. Since the algorithm generates candidate feature sets using multiple criteria and all feature classes jointly, it is able to capture most of those features which are discriminating.

The following are the feature selection criterias we used and implemented in EFS:

- **Information Gain** (IG): based on entropy.

$$IG(f) = -\sum_{i=1}^{m} P(c_i)logP(c_i) + \sum_{f,\bar{f}} P(c_i|f)log(P(c_i|f)$$

- **Mutual Information** (MI): metric that is commonly used in statistical language modeling.

$$MI(f,c) = \sum_{f,\bar{f}}\sum_{c,\bar{c}} P(f,c)log\frac{P(f,c)}{P(f)P(c)}$$

- **χ^2 Statistic**: measures the lack of independence between a feature and class.

$$\chi^2(f,c) = \frac{N(WZ-YX)^2}{(W+Y)(X+Z)(W+X)(Y+Z)}$$

- **Cross Entropy** (CE): this metric is similar to mutual information.

$$CE(f) = P(f)\sum_{i=1}^{m} P(c_i|f)\log\frac{P(C_i|f)}{P(f)}$$

- **Weight of Evidence for Text** (WET): the criterion is based on the average absolute weight of evidence.

$$WET(f) = \sum_{i=1}^{m} P(c_i)P(f)|log\frac{P(c_i|f)(1-P(c_i))}{P(c_i)(1-P(c_i|f))}$$

3.7 Textstat

We leverage a library named textstat that calculates statistics on text to determine readability, complexity, and grade level of a particular corpus. [10] In particular, we extract and use two features that were found to be useful in the gender classification task.

1. Lexicon Count: number of words present in the text, whist removing punctuation.
2. Readability Consensus: the estimated school grade level required to understand the text.

3.8 Spelling

We leverage a library named pyspellchecker based on Peter Norvig's spell checking algorithm. [11] [12] It uses a Levenshtein Distance algorithm to find permutations within an edit distance of 2 from the original word. [13] It then compares all permutations to known words in a word frequency list. Words that are found more often in the frequency list are more likely the correct results.

Using pyspellchecker, we iterate through a single text corpus and count the amount of misspellings that are con-

tained in that particular text. In addition to its standard word library, we add additional words that we deem "valid" manually. Table 3 shows the additional words added.

Category	Words
Puntuations	"'s", "'m", "'re", "'ll", "'ve", "'t", "'d"
Words	'hulu', 'google', 'smartphone', 'iphone', 'immersive', 'xbox', 'blog', 'obama', 'facebook', 'itunes', 'gmail', 'espn', 'youtube', 'adsense', 'pikachu', 'etsy', 'wikipedia', 'starbucks', 'jetblue', 'webcam', 'traveler', 'retweet', 'website', 'favors', 'deadspin', 'huffington', 'wordpress', 'linkedin', 'website', 'mozilla', 'firefox', 'linux', 'http', 'evanescence', 'dvd', 'krispy', 'kreme', 'donut', '3d', 'mp3', 'jpg', 'photobucket', 'suv', 'saudia', 'vevo', 'popsicle', 'voicemail', 'sophomore', 'hotmail', 'fastmail', 'morrowind', 'fanboy', 'fandom', 'resize', 'ie5', 'ie6', 'timeline', 'newbie', 'vigor', 'netflix', 'anime', 'html', 'xml', 'savior', 'walgreens', 'ebay', 'myspace', 'lasik', 'wii', 'uber', 'petco', 'sweatpants', 'jello', 'malware', 'wiki', 'ubuntu', 'savor', 'nerdy', 'flavors', 'jalapenos', 'guestbook', 'runescape', 'ragnarok', 'gamer', '2d', '3d', 'neverwinter', 'paintball', 'urllink', 'internship', 'warcraft', 'simcity', 'sxsw', 'paralyzed', 'telus', 'shrek', 'avatar', 'poptarts', 'bioshock', 'endeavor', 'walmart', 'multiplayer', 'advil', 'ffix', 'funimation', 'fullmetal', 'dvds', 'blogs', 'blogging', 'blogger', 'bloggers', 'retweets', 'iphones', 'tweets', 'websites', 'resizes', 'resizing', 'geeks'

Table 3 – Additional spelling vocabulary.

3.9 Feature Value Assignment

Values are assigned differently to different features. There are four common ways of feature value assignments: Boolean, Discrete, TF (Term Frequency), and TF-IDF (Product of Term and Inverted Document Frequency). The boolean scheme assigns a 1 to the feature value if the feature is present in the document and 0 otherwise. The discrete scheme assigns an integer count value if possible and converts continuous values to discrete ordinal values. The TF scheme assigns the relative frequency of the number of times that the feature occurs in the document. The TF-IDF scheme assigns the value of a feature using the formula:

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D)$$

Where t is the term or feature, d is the document, and D is the entire document set.

Through preliminary experiments, we found that TF-IDF yielded the best results, so for most of our results we will be using the TF-IDF scheme unless otherwise stated.

For our best results, the feature value assignments for each feature is as follows:

- **F-measure**: Actual Value, Min/Max Scaled
- **Stylistic Features**: TF-IDF
- **Gender Preferential Features**: TF-IDF
- **Factor Analysis**: TF-IDF
- **POS Sequence Patterns**: TF-IDF
- **Word Count**: Actual Value, Min/Max Scaled
- **Textstat**: Actual Value, Min/Max Scaled
- **Spelling**: Actual Value, Min/Max Scaled

We also try different feature value assignments however they give suboptimal results so we won't go into detail of their feature value assignments.

4 Novel Method

Although A. Halevy et al. argues that introducing more data for some tasks have positive effects on classifiers, we found through experiments that aimlessly adding more labeled data had detrimental effects on the classification accuracy (if we added too much). We found that adding a little bit of data from Dataset 2 to Dataset 1 had a decent improvement to accuracy, however adding too much had detrimental effects. Therefore, we chose to add 3000 labeled data to our total dataset.

In adding that data, our accuracy improved from approximately 72% accuracy to 75% accuracy (10-fold cross validation). However, we still had approximately 15,000 labeled blog posts for us to leverage.

4.1 Supervised Semi-Supervised Learning

Since we know that adding too much more data has a detrimental effect on our classifiers, we want to only add blog posts that will strengthen our classifier's total accuracy.

To leverage the remaining labeled data that we have, we now present the proposed Supervised Semi-Supervised (SSS) Learning algorithm. This algorithm is very much based on the concepts of Semi-Supervised Learning. We want to be able to add more data into our training data set such that it improves classifier accuracy. It is essentially Semi-Supervised learning, and what makes it "supervised" is the fact that we validate the inferred classification with the known classification. Our Supervised Semi-Supervised Learning algorithm is given below in Figure 1.

Algorithm 1 SSS_Learning(*training_data*, *blog_data*)

```
max_iter ⇐ number of iterations
max_added ⇐ number of added blog posts per step
iteration ⇐ 0
while iteration < max_iter do
    iteration++
    for i = 0 to 10 do
        data ⇐ 10% split of training_data
        clf ⇐ Classifier(data)
        prob_male, prob_female ⇐ clf.PredictProbability(blog_data)
        for j = 0 to max_added do
            blog_post ⇐ Using random weighted choice based on highest
            prob_male, pick a blog post
            if blog_post.class = male then
                Add blog_post to training_data
            end if
        end for
        for j = 0 to max_added do
            blog_post ⇐ Using random weighted choice based on highest
            prob_female, pick a blog post
            if blog_post.class = female then
                Add blog_post to training_data
            end if
        end for
    end for
end while
return training_data
```

Figure 1 – SSS Learning Algorithm

We now briefly explain the SSS Learning algorithm. The algorithm takes in input of $training\_data$ and $blog\_data$. It will then loop based on the max_iter value. 10% of the data will be split from the $training\_data$ and then be used to build a classifier. This classifier will then be used to get probability predictions from the $blog\_data$. Based on the probabilities, we then randomly choose max_added blog posts using a weighted algorithm (higher probabilities have higher chance of being selected). The reason we do a weighted random selection to try and avoid data overfitting.

We then check if the blog post has the same class as the predicted class. If it is, then we add the blog post into $training\_data$. We do this for male and female and repeat for max_iter. In our experiments we used $max\_iter = 1$ and $max\_added = 200$.

5 Experimental Results

This section evaluates the proposed techniques and sees how they affect the classification accuracy. Before applying SSS Learning, all results are from 10-fold cross validation. After SSS Learning, all results are average accuracies on different validation sets from multiple runs.

We note that Ensemble Feature Selection did not yield good results in terms of classification accuracy, and had high resource demands (RAM) and high execution times (in fact, it was at times detrimental), and therefore we do not use it unless specifically stated.

5.1 Classifiers

We used many classifiers throughout our experiments including Naïve Bayes, SVM, SVM-R, Bagging, Bagging-R, and Neural Networks (MLP and Keras). We found Linear SVMs and Neural Networks to produce the best results. Due to the heavy computational requirements, high execution times, and fluctuating final accuracies of Neural Networks, we use SVMs in most cases. This includes our novel Supervised Semi-Supervised Learning algorithm.

The hidden layers used in our MLP classifier were 100, 50, 25. The layers that we use for our Keras classifier were 100, 25, 50, with two 0.3 dropouts to avoid overfitting.

5.2 Pipeline

A general overview on how we produce our results from running our program are as follows:

- Dataset 1 is tested via 10-fold cross-validation as a baseline.
- A new dataset of Dataset 1 and 3000 records of Dataset 2 will be created.
- A 90/10 training/validation set will be extracted from the new dataset. This validation set is set aside and never looked at by any of our models. If our models need some type of testing dataset to work off of, we split it from the training dataset.
- Supervised Semi-Supervised Learning algorithm will be run on the training dataset. The training dataset will be expanded in this step.
- Various classifiers will be run on the resultant training dataset and validation dataset.

5.3 Baselines

To get a baseline on the accuracies, we ran various classifiers using their default hyper-parameters and used a simple bag-of-words vectorizer without any preprocessing. Table 4 shows the baseline results. We also includee A. Mukherjee et al. results [1] as one of the baselines.

Classifier	Accuracy
Naïve Bayes	66%
Decision Trees	65%
Logistic Regression	59%
Gradient Boosting	65%
Random Forest	57%
A. Mukherjee et al. [1]	88.56%

Table 4 – Baseline accuracies using Dataset 1

5.4 Results

Table 5 shows the metrics after performing 10-fold cross-validation on Dataset 1 + 3000 records from Dataset 2 using a Linear SVM classifier. We see a ~3% increase of accuracy from adding 3000 records from Dataset 2. We do not include cross-validation accuracies on the training dataset after SSS Learning as the dataset was "seen" by the model and no longer valid for cross-validation scores. For interested readers, we note that the 10-fold cross validation classification accuracy on the training dataset post-SSS Learning is ~80%.

Data	Accuracy	Precision	Recall	F-score
Dataset 1	72.09%	72.20	72.09	72.09
Dataset 1 + Dataset 2 (3000)	75.21%	75.33	75.21	75.20

Table 5 – Linear SVM, 10-fold cross-validation scores.

Table 6 shows an example of accuracy increases on a validation set as SSS Learning is run. The final accuracy is noteably much better than the initial accuracy (~1.5% increase). However, we can see that the accuracy fluctuates up and down as each iteration is run. We note that there could be possible optimization potentials that optimally pick the blog data to add without peaking at the validation data. (Note: Our algorithm does not "see" the validation accuracies as it runs, it is just displayed for our information)

Iteration in SSS	Accuracy
Initial	74.4%
1	75.2%
2	74.9%
3	74.6%
4	75.7%
5	75.4%

6	75.6%
7	75.6%
8	75.4%
9	76.7%
Final	**75.9%**

Table 6 – Linear SVM, SSS Learning accuracy improvements.

Table 7 shows the accuracies of the best classifiers before and after SSS Learning. We can see that our SSS Learning algorithm has a positive effect on the final classification accuracy. The overall best accuracy we were able to achieve using our method was 77.5% using a Multi-layer Perceptron Classifier post-SSS Learning.

SSS	Classifier	Accuracy
Pre	MLP	76.4%
Post	MLP	**77.5%**
Pre	Keras	75.4%
Post	Keras	**76.4%**
Pre	SVM	74.4%
Post	SVM	**75.9%**

Table 7 – Accuracy comparisons of Pre and Post SSS Learning.

Table 8 shows the various other accuracies of classifiers that we tried out on the final data set. We can see that SVM/SVM-R performed the best when compared to Naïve Bayes and Bagging. It was not, however, able to perform better than the neural networks in Table 7. We also attempted a majority voting to decide on a final classification based on all of the classifiers we ran. This ended in a sub-optimal classification accuracy. In general, we believe this to be due to all the classifiers being in "agreement" when picking male or female.

Classifier	Notes	Accuracy
Naïve Bayes	TF-IDF, EFS	65.4%
Naïve Bayes	Discrete, EFS	63.0%
Naïve Bayes	Bool, EFS	67.7%
SVM (Linear)	Bool	71.6%
SVM (Linear)	TF-IDF, EFS	72.3%
SVM (Linear)	Discrete	73.4%
SVC (Linear)	TF-IDF	72.8%
SVM-R (Linear)	TF-IDF	74.3%
SVM-R (RBF)	Bool	72.1%
SVM-R (RBF)	TF-IDF	72.5%
Bagging (DT)	Bool	68.0%
Majority Vote	Includes MLP/Keras	73.8%

Table 8 – Accuracy of various other classifiers that we tested our final dataset on.

6 Conclusion and Future Work

This paper and its implementation studied the problem of gender classification. Although there have been several existing papers studying the problem, the current accuracy is still far from ideal. We implemented various features from many different works including POS Sequence Pattern Mining and Ensemble Feature Selection. In this work, we followed the supervised approach and proposed a novel technique to improve the current state-of-the-art. In particular, we proposed a new learning technique named Supervised Semi-Supervised Learning which leverages additional labeled data. Since we have so much labeled data, it is important to find a subset of data that have positive effects on the classification task. This novel method helps us achieve higher classification accuracy.

For future works we would like to explore optimizations of our novel method such that any detrimental datasets are not added to the final result. We would also like to explore in more depth as to why Ensemble Feature Selection was not as useful in our project as it was in the paper by A. Mukherjee. et al. [1]

Acknowledgment

We wish to thank Dr. Mukherjee of the University of Houston Department of Computer Science for his continued support throughout this project.

References

[1] A. Mukherjee, and Bing Liu, "Improving Gender Classification of Blog Authors" Proceedings of the 2010 Conference on Emperical Methods in Natural Language Processing. (2010) Available: https://www.aclweb.org/anthology/D10-1021

[2] A. Halevy, P. Norvig, F. Pereira. "The Unreasonable Effectiveness of Data" IEEE Intelligent Systems.

[3] Schler, J., Koppel, M., Argamon, S, and Pennebaker J. "Effects of age and gender on blogging" In Proc of the AAAI Spring Symposium Computational Approaches to Analyzing Weblogs. (2006)

[4] Argamon, S., Koppel, M., Pennebaker, J. W., Schler, J. "Mining the Blogosphere: Age, Gender and the varieties of self-expression" First Monday (2007)

[5] Yan, X., Yan, L. "Gender Classification of Weblog Authors" Computational Approaches to Analyzing Weblogs, AAAI. (2006)

[6] Nowson, S., Oberlander J., "Gender Genres, and Individual Differences" In Proceedings of the 27th annual meeting of the Cognitive Science Society (2005)

[7] NLTK. [Online] Available: https://www.nltk.org/

[8] Sample Blog Dataset. [Online] Available: http://www.cs.uic.edu/~liub/FBS/blog-gender-dataset.rar

[9] Blog Authorship Corpus. [Online] Available: http://u.cs.biu.ac.il/~koppel/BlogCorpus.htm

[10] Textstat. [Online] Available: https://github.com/shivam5992/textstat

[11] Pyspellchecker. [Online] Available: https://github.com/barrust/pyspellchecker

[12] P. Norvig. "How to write a spelling corrector" [Online] Available: https://norvig.com/spell-correct.html

[13] Levenshtein Distance. [Online] Available: https://en.wikipedia.org/wiki/Levenshtein_distance

[14] Flair. [Online] Available: https://github.com/zalandoresearch/flair