Twitter User Gender Classification

Boyi He BOYIHE@SEAS.UPENN.EDU
Chenchen Hu HU10@SEAS.UPENN.EDU
Xuye Lin XUYLIN@SEAS.UPENN.EDU

1. Introduction

Gender classification with high accuracy has many useful applications in marketing, advertising, public relation management and criminal investigation. Social media such as Facebook and Twitter have been gathering personal information and preferences to provide personalized user experience. Our goal in this project is to find out the difficulty of training a model to predict a person's gender based on their Twitter profile and random tweets.

Historically gender classification based on twitter profiles has been addressed from different perspectives, including user tweets, user main page color, user emojis, user posted images and so on (Burger et al., 2011; Merler et al., 2015; Alowibdi et al., 2013). Among them, language models and image-based models give really impressing accuracy around 80% to 90%. In this project, the only readily available public data set we have found is the one provided by CrowdFlower. The data set contains a mix of several types of features including URLs to profile images, texts, numeric values and hexadecimal colors. The nature of the data set brought threats to the validity of our study. The first challenge was the validity of the labels. Since Twitter users' gender information was given, CrowdFlower asked a group of contributors to label each user 'male', 'female' and 'brand' with a confidence value. A complete evaluation or verification of all the 20,050 user gender would be impractical. By using the labels provided by contributors we were making a compromise on the quality of the labels. The prediction of our model could only be as good as one person's gender observed by another. To address this issue we manually checked 300 out of the 20,050 profiles. In all cases where the confidence value was higher than 0.6, we believe that the gender labels given by the contributor are correct.

The second challenge was that the actual size of meaningful data was much smaller than we expected. Instances labeled 'brand' had no use for our purpose and with those instances removed, we were left with only two-thirds of the original size. What's more, we realized that the useful features we selected might not be all that useful. For example, retweet counts mostly had value 0 and their distribution was heavily skewed. A large number of color feature values were not valid hexadecimal RGBs. The URLs to profile images were mostly expired. However, we believe the focus of our study is not to improve model accuracy by enlarging the current data set. Rather, we should utilize the existing data as much as possible.

2. Feature Selection

2.1. Data Preprocessing

Table 1. Features and Missing Values

Feature	Type	Missing and Invalid
gender	string	7156
gender:confidence	float	26
fav_number	int	0
link_color	hex	689
tweet_count	int	0
retweet_count	int	35
sidebar_color	hex	651
description	text	3744
text	text	0
screnn name	string	0
profile image	url	12651

Our data set contains 20,050 rows with 26 features. We selected 9 meaningful features and discarded those that we thought had no correlation with gender, including geolocation and timezone. We also discarded retweet counts because the distribution was heavily skewed. The result of running models with and without this feature showed that removing retweet counts did not have impact on model performance.

Before looking into various features, we removed 7156 rows that couldn't be classified as male or female. Based on the result of manual checking mentioned above, we also eliminated samples that had low confidence values (gender confidence < 0.6) in our first round of training. Then we used fine-tuned models to predict the results of removed samples and compared them with the original labels. If the result matches, we added them back. In the final stage, we retrained the models with the new data set. We believe this

approach is a good compromise that allows us to utilize the data to maximum extent without introducing too much noise.

For text features, we focused on three fields, users' tweets ("text"), self introductions ("description") and screen name ("name"). Firstly we filled missing values with empty strings, then we removed non-ASCII characters, URLs, tweet tags, tweet mentions and emojis from text and converted words to lower case. To transform text into numerical features, we leveraged the bag of words model and explored among TF (Term Frequency), Binary TF and TF-IDF(Inverse Document Frequency) representations. After comparing their cross-validation performance, we selected binary TF (1 if token exists in the text, 0 otherwise) as the final text feature. Specifically, we followed John D's work (Burger et al., 2011): used English stop words and tokenized tweets and description into 1-2 grams words and 1-5 grams characters as for screen names. These parameters were latter proved to be golden in our experiments. Surprisingly, setting a maximal number of word features would reduce the testing accuracy (-6%). Therefore the final text features consisted of 30000+ dimensions.

We have also chosen three types of non-text features: integer, hexadecimal RGB colors and URLs to images. Integer features are favorite numbers, tweet and retweet counts. These mixed data types each requires different cleaning and conversion, which might depend on the model we selected. For example, models that exploit distances or similarities between samples such as SVM require feature scaling.

Color features represent link colors or sidebar colors. Initially we only removed invalid values that contained special characters possibly due to encoding issue. However, valid RGB colors in HTML should be represented by hexadecimal triplets. When we applied a more strict regex pattern for checking, over 1000 samples failed the test. We selected a few of these invalid instances and looked into the HTML source of the corresponding users' profile page. It became clear that the values were correct except they were missing 0s on the left. We zero-padded these values and filled them back in. For the 42 instances that contained special characters, we replaced them with the mode values of their column.

Lastly, for the image features, we attempted to download all the profile images with the URLs, but more than half of the links were already unreachable. To filter out unqualified images, firstly, we filtered out the images with brand in the gender attribute, i.e. not an account for an individual. Secondly, we noticed that the downloaded data contained 4 versions of default twitter profile images. To filter out those, we loaded the RGB representation array of each image and compute the distance (l_2 norm) between it and each of the 4 default images. If there was a close match (very small distance), we would label that image as a default image. Furthermore, some images have a slightly different shape (58 x 48), so we clipped them into the standard shape. After filtering, we had 3,093 images available. Then we load the images as arrays in both grayscale and RGB for different models.

3. Method

3.1. Classification on Words

To evaluate the prediction power of different text features, we first trained a multinational Naive Bayes model on "text", which generated 60% accuracy; while using "description" gave 65% accuracy and 71% for "name". After appending these features, we got a model of 73% accuracy. We experimented with four major classifiers for all three text features: Multinomial Naive Bayes, SVM with cosine similarity kernel, logistic regression and MLP (MultiLayer Perceptron). Previous works (Burger et al., 2011) has shown that Balanced Winnow2 was the most powerful classifier for tweets. After comparing the implementation details, we chose MLP [5000,2000,100,2] as a comparable substitution and got relative low accuracy (54.49%). We believed it was due to overfitting and sparsity of count vectors. For other classifiers, we searched for the best parameters by leveraging gridsearch and used 10-fold cross validation to evaluate the classifier performance. The performance metrics are as shown in Table 1.

In the second round, we added back 200+ samples from the unused dataset and retrained the model. As shown in the table, the test accuracy increased by 0.5%

Table 2. Text Features Classification Performance First Round

CLASSIFIER	ACCURACY	PRECISION	RECALL	TIME (S)
MULTINB(1ST)	0.7293	0.7230	0.7156	0.0210
MULTINB(2ND)	0.7351	0.7220	0.7254	0.0217
SVM	0.7002	0.6710	0.7164	18.5693
LOGISTIC	0.7214	0.7034	0.7172	0.3271

3.2. Other non-text features

Table 3. Non-text Features Classification Performance First Round

CLASSIFIER	ACCURACY	+/-STD	PRECISION	+/-STD	RECALL	+/-STD	TIME (S)
LOGISTIC REGRESSION	0.566	0.0156	0.538	0.0133	0.629	0.0264	0.1933
RANDOM FOREST	0.591	0.0195	0.568	0.0107	0.555	0.0093	0.9423
SVM-RBF	0.551	0.0210	0.526	0.0177	0.661	0.0386	15.4630
GAUSSIAN NAIVE BAYES	0.542	0.0132	0.525	0.0098	0.669	0.028	0.0427

We trained our models separately on four other non-text features ('fav_number', 'link_color', 'sidebar_color','tweet_count'). Tweet counts and favorite

Table 4. Non-text Features Classification Performance Second Round

CLASSIFIER	REUSED	ACCURACY	+/-STD	PRECISION	+/-STD	RECALL	+/-STD
LOGISTIC REGRESSION	218	0.566	0.0134	0.538	0.0110	0.629	0.0235
RANDOM FOREST	198	0.591	0.0102	0.568	0.0055	0.555	0.0060
SVM-RBF	160	0.551	0.0187	0.526	0.0161	0.661	0.0386
GAUSSIAN NAIVE BAYES	215	0.542	0.0138	0.525	0.0105	0.669	0.0280

number are not normally associated with one's gender. Most online machine learning projects built with this data set have discarded these features as well. Nevertheless we'd like to see if these features have any use at all. After experimenting with some quick and dirty models, we selected four most promising ones: Logistic Regression, Random Forest, SVM with rbf, and Gaussian Naive Bayes. The hyper-parameters were tuned using grid search. We used 3-fold cross validation to evaluate the performance. The results are given in Table 3.

After comparing the prediction and the original labels, we added back some of the removed samples with confidence value less than 0.6. We retrained the models on the new data set. The result as well as the number of reused samples are shown in Table 4. Execution time is omitted from Table 4 since with a couple hundred more samples the difference can be ignored.

In terms of accuracy, the performance given by all four models was slightly less than 60%. The performance of the second round has improved by a small percentage. A comparison between the models' training and test accuracy showed that the models were not over-fitting. We could only conclude that either there was too much noise in these columns or the features were not relevant at all. As is shown by Table 5 and 6, the top five frequent values of these features did not discriminate genders.

Table 5. Top 5 frequent values: favorite number and tweet counts

FAVOURITE NUMBER				TWEET COUNT			
Male		Female		Male		Female	
value	%	value	%	value	%	value	%
0	2.29	0	3.31	3	8	76289	14
1	0.71	1	0.55	8	7	79641	8
2	0.42	2	0.41	4	6	11	8
3	0.37	3	0.36	114	5	8172	6
5	0.35	5	0.22	20898	5	97	5

3.3. Image Classification

The profile images could have strong prediction power, since the profile image could be a photo of the user, or an image that have strong indication of gender. The first model that we applied was PCA and SVM with a Gaussian kernel from scikit-learn. We set the number of Eigenfaces of PCA to be 150 and used GridSearchCV to search for the optimal C (penalty) and gamma for SVM, which provided an accuracy of 57.8%.

A main reason for PCA not to success could be the low resolution of the images that we found (48 x 48). Attempting to obtain better training images, we tried to crawl the profile images with an original size via the Twitter API. However, since the training set was updated a year ago, a large portion of usernames were out of date that we could not retrieve their profile images. Also, our crawler was restricted by the query rate limitation of Twitter API, so we were not able to obtain large set of images with higher resolutions.

With a low resolution, the face recognition models that were already available to us, such as Viola-Jones Algorithm and the Dlib state-of-art face detector built with deep learning, did not work well. We expected that with higher resolution images, we could extract images with faces detected. Running PCA with only those images would provide us a better result on classification.

Secondly, we experimented with Convolutional Neural Networks (CNN). We referred to the architect of VGG network, used (conv2d, batchnorm, relu) as a block, maxpooling between blocks, followed by fully connected layer after 8 blocks and cross entropy loss as the only loss layer. The parameters of network is shown in the table 7. We trained the model for 5 epochs, for each of which we randomly shuffled the images and iterated 6,186 steps (each image was processed 2 times). The resulting accuracy was 59.9%. CNN did not performed well as it failed to converge through the training. We believe that since the images had too much variety (photos, random graphics), the model did not manage to find common features among the pictures.

Table 6. Top 5 frequent values: link and sidebar colors

LINK COLOR				SIDEBAR COLOR			
Male		Female		Male		Female	
value	%	value	%	value	%	value	%
0084B4	23.96	0084B4	20.57	C0DEED	21.48	C0DEED	19.26
9999	2.20	9266CC	2.19	0	9.40	FFFFFF	11.71
3B94D9	1.99	F5ABB5	2.19	FFFFFF	8.93	0	11.57
2FC2EF	1.56	DD2E44	1.40	EEEEEE	2.09	EEEEEE	1.63
DD2E44	1.38	FF0000	1.30	181A1E	1.24	65B0DA	0.95

3.4. Stacking Models

We integrated the best performing models of three feature categories by soft voting: weighted their predictions by the corresponding accuracy scores. The performance is shown in table 8. As expected, the text model defeats

Table 7. Covolutional Neural Network Parameters

LAYER	KERNEL SIZE	CHANNEL	PADDING	STRIDE
CONV2D$_1$	5	64	2	1
CONV2D$_2$	5	64	2	1
MAXPOOL$_1$	2	1	0	2
CONV2D$_3$	5	128	2	1
CONV2D$_4$	5	128	2	1
MAXPOOL$_2$	2	1	0	2
CONV2D$_5$	3	256	1	1
CONV2D$_6$	3	256	1	1
MAXPOOL$_3$	2	1	0	2
CONV2D$_7$	3	128	1	1
CONV2D$_8$	3	128	1	1
FC		4608X1		
SIGMOID				

the stacking model, probably because other non-text models are highly overfitted and introduce more variance to the stacking model through linear combination.

Table 8. Stacking Model Performance Matrix

FEATURE	MODEL	TRAIN ACC	TEST ACC
TEXT	MULTINB	0.9234	0.7351
COLOR+NUMBER	RANDOMFOREST	0.9863	0.591
IMAGE	CNN	0.9897	0.5997
ALL	SOFT VOTE	0.9797	0.7327

4. Conclusion

Text features, among all relevant fields such as number, color and profile image, give best predicting result for gender classification. Previous studies have shown that users' true name has strongest correlation with gender, followed by user's screen name(Burger et al., 2011). This is again demonstrated in our experiments that screen name (71%) outperforms the concatenated features generated by both description and text (68%). Further more, we observed a large discrepancy between training and testing accuracy. We assume it's resulted from the limitation of available tweets per user. According to John's work shown in figure 1, the accuracy linearly increases as tweets from the target user goes up(2011). Small amount of training corpus gives extreme sparse feature matrices and slight correlations between users. Therefore the classifier couldn't draw generalizable boundary in this high-dimensional and sparse parameter space. As for model selection, traditional linear classifiers such as SVM and logistic regression perform alike. We chose multinomial Naive Bayes because of its short training time and relative immunization to overfitting.

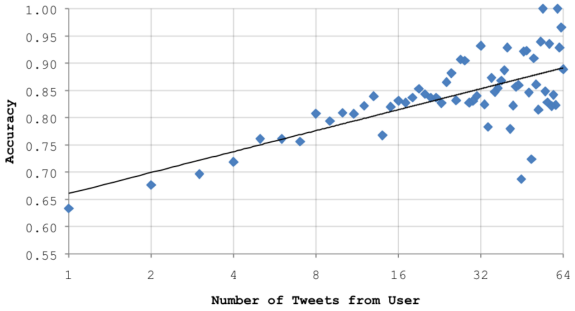


Figure 1. Performance increases with more tweets from target user

5. Future Works

As mentioned in Conclusion, larger number of tweets per user will greatly boost the performance. We propose to crawl more user profile images and tweets if time allowed and try dimension reduction methods such as PCA to make text features denser.

Another regrettable part of the project was the unsatisfactory result of image classification. Although we adopted classical deep neural network architecture from VGG, without image normalization, random flipping, random cropping and other reprocessing, CNN wouldn't be able to extract representative features. Future works could be done to use pretrained networks (such as VGG19) to guide the CNN training process. Also we suggest to classify user profile images into general categories such as adult, nature, sports and so on by using leading edge CNNs, and then vectorize image based on their category information (Merler et al., 2015)

References

Alowibdi, Jalal S, Buy, Ugo A, and Yu, Philip. Language independent gender classification on twitter. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 739–743. ACM, 2013.

Burger, John D, Henderson, John, Kim, George, and Zarrella, Guido. Discriminating gender on twitter. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1301–1309. Association for Computational Linguistics, 2011.

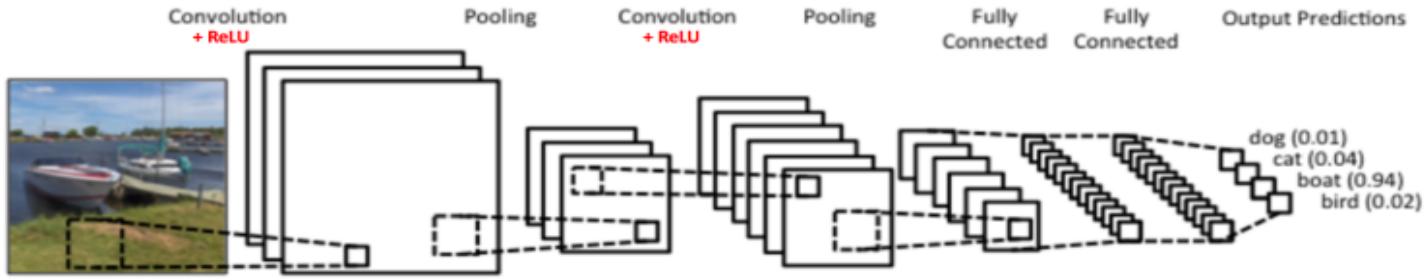
Merler, Michele, Cao, Liangliang, and Smith, John R. You are what you tweet pic! gender prediction based on semantic analysis of social media images. In *Multimedia and Expo (ICME), 2015 IEEE International Conference on*, pp. 1–6. IEEE, 2015.

Twitter User Gender Classification

Boyi He, Chenchen Hu, Xuye Lin

Unlike Facebook, Twitter does not ask users to specify their gender. Our goal in this project is to find out the difficulty of training a model to predict a person's gender based on their Twitter profile and random tweets. It's wide applied in personalizing user experience, advertising and public relation management.

Feature	Preprocessing	Feature
User Description	Remove non acsii code	1-2 word grams binary countVector
User tweets (one)	Remove tag, at, emoji, http link	1-2 word grams binary countVector
User Screen Name	Split by non alphabetic chars	1-5 char grams binary countVector
Profile sidebar color&link color	Validate hex color string using regex	numeric value
Profile Image (7399)	Remove default image, broken links	Batch size = 1, 48x48 RGB image



Features	Model	Accuracy
Text Features	SVM-cosine	70.02%
	MLP	54.49%
	MultiNB	**73.51%**
Color Features	Logistic	56.6%
	Random Forest	59.1%
	SVM-rbf	55.1%
	Gaussian NB	54.2%
Profile Images	PCA + SVM-Gaussian	57.8%
	CNN	59.9%

Conclusions

Text Features

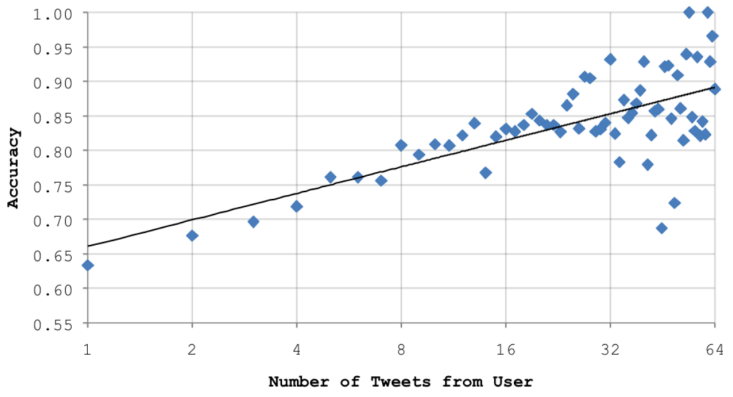
- User Screen Name is the most distinguishing feature
- Tweets per user vs Accuracy: table 1 shows that one tweet per user can achieve as much as 66% accuracy (bottleneck)
- Multinomial Naïve Bayes performs best among other models

Color Features

- The most promising model is random forest
- The tables 2-3 show that non-text features are irrelevant to gender. Therefore it's hard to train a satisfying model

Image Features

- Small size of images (48 x 48), which limited the ability for applying current face recognition models to filter out non-human pictures
- Crawling larger size of images from Twitter APIs could help



FAVOURITE NUMBER				TWEET COUNT			
Male		Female		Male		Female	
value	%	value	%	value	%	value	%
0	2.29	0	3.31	3	8	76289	14
1	0.71	1	0.55	8	7	79641	8
2	0.42	2	0.41	4	6	11	8
3	0.37	3	0.36	114	5	8172	6
5	0.35	5	0.22	20898	5	97	5

Top 5 frequent values: favorite number and tweet counts

LINK COLOR				SIDEBAR COLOR			
Male		Female		Male		Female	
value	%	value	%	value	%	value	%
0084B4	23.96	0084B4	20.57	C0DEED	21.48	C0DEED	19.26
9999	2.20	9266CC	2.19	0	9.40	FFFFFF	11.71
3B94D9	1.99	F5ABB5	2.19	FFFFFF	8.93	0	11.57
2FC2EF	1.56	DD2E44	1.40	EEEEEE	2.09	EEEEEE	1.63
DD2E44	1.38	FF0000	1.30	181A1E	1.24	65B0DA	0.95

Top 5 frequent values: link and sidebar colors