CS5740: Assignment 1

`https://github.com/cornell-cs5740-21sp/a1--group-5`

Kae-Jer (Mike) Cho kc2225
Vianne Gao vrg36
Yuxin Zhang yz2726

1 Introduction (5pt)

We implement the Perceptron model and the multi-layer Perceptron (MLP) model and apply them to two taks: Proper Name Classification where we take proper names and predict whether they indicate people, movie, drugs, place and or company; and Newsgroup Classification where we classify documents into 20 topics.

We experiment with different input features for each dataset and model separately, perform ablation tests on features to identify the most informative inputs, and experiment with different model complexity, activation functions and optimizers for the MLP model. Overall, we observed that MLP outperforms Perceptron in both tasks, and both models are generalizable to unseen dataset.

2 Features (20pt)

There are three feature components in both propername data feature and newsgroup data feature when doing feature extraction (N-gram, Task-specific features, and combined features).
For the task-specific feature for newsgroup dataset, we engineered 4 features based on the following observation:

- Each group is distinguished by whether headers such as NNTP-Posting-Host: and Distribution: appear more or less often. Feature **host_count**: count the num. of "NNTP-Posting-Host:". Feature **distribution_count** : count the num. of "Distribution:".
- Another significant feature involves whether the sender is affiliated with a university, as indicated either by their headers or their signature. Feature **from_university**: binary variable for whether email address of sender contains "edu".
- The word "article" is a significant feature based on how often people quote previous posts like: "In article [article ID], [name] <[e-mail address]> wrote:". Feature **reference_count**: count the num. of "In article" in text.

For the task-specific feature for propername dataset, we engineered 4 features based on the following observation:

- Text containing numbers can hardly be names. Feature **contain_numerics**: binary variable whether text contains numbers.
- Text containing some punctuation (-:"%) can hardly be names. Feature **contain_special_punc**: binary variable whether text contains special punctuation (&-:"%).
- Text containing "Inc." can be strong indicator for the class company. Feature **contain_inc**: binary variable whether text contains "Inc." in the end.
- Text containing only one token can hardly be a proper name. Feature **small_token_length**: binary variable whether text contains only token length 1.

	Propername dataset	Newsgroup dataset
N-gram	Character-level N-gram feature engineering, controlled by N and MinFreq(see text)	Bag of words model Feature engineering controlled by N and MinFreq(see text)
Task-specific	See feature definitions above	See feature definitions above
Combined	Horizontally stack N-gram features and task-specific features	Horizontally stack N-gram features and task-specific features

Table 1: Feature components

The word and character frequency dictionary is built on the text from the entire dataset to maximize the inclusion of features and avoid unknown words or characters. In our N-gram models, we allow input of N to be a tuple specifying what range of word or character sequence should the feature engineering model consider. MinFreq (minimal Frequency) is introduced for the feature engineering process to select the most prominently appeared word or character, thus reducing dimensionalities.
Table 2 shows an example of how the generated three feature components look like on the propername datasetset with N = (1,1) and MinFreq = 15.

3 Experimental Setup

Data (5pt) We divide the Proper Name dataset into training, development and test sets. The

N-gram feature

	m	a	l	h	t	o	s	b	r	w	...	1	5	9	x	6	4	3	8	7	Label
0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	place
1	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	person
2	0.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	drug
3	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	drug
4	1.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	person

Task-Specific Feature

	contain_numerics	contain_special_punc	contain_inc	Small_token_length	Label
0	0.0	0.0	0.0	1.0	place
1	0.0	0.0	0.0	0.0	person
2	0.0	0.0	0.0	1.0	drug
3	0.0	0.0	0.0	1.0	drug
4	0.0	0.0	0.0	0.0	person

Combined

	m	a	l	h	t	o	s	b	r	w	...	8	/	7	!	*	contain_numerics	contain_special_punc	contain_inc	Small_token_length	Label
0	1.0	1.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	place
1	1.0	1.0	0.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	person
2	0.0	1.0	1.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	drug
3	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	1.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	drug
4	1.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	person

Table 2: Demonstration of feature components

Statistics	Proper Name	Newsgroup
Num. Training	23121	9051
Num. Development	2893	2263
Num. Testing	2862	7532
Sample Length Mean	14.2	283.66
Sample Length Std.	7.80	520.33
Vocab Size	149	386410

Table 3: Dataset Summary Statistics

number of samples, name/document length distributions and vocabulary size are summarized in Table 3. Concretely, the vocabulary size for Proper Name is the number of unique characters across all samples, and for 20Newsgroup is the number of unique words (instances separated by empty spaces) across all samples.

Data Preprocessing (5pt) We seek different approaches for text data preprocessing in the newgroup dataset and the propername dataset. This is because we believe that newsgroup text contains long sentences of which the classification is decided by semantic meaning of each word or sequence token. That's why we removed punctuation and stopwrods from it. Yet for the propername text, since it's composed by rather shorter character combinations, we keep the punctuations which might be indicative for some classes in order to preserve feature complexity. The detailed preprocessing are as below
For newsgroup dataset:

- Convert text to lower case
- Remove url in text
- Remove non-ascii in text
- Remove punctuation (no need for propername dataset)
- Remove stopwords (no need for propername dataset)

Perceptron Implementation Details (5pt) We implement a multi-class Perceptron that takes the dense representation of a document $\bar{d}$ as input where $\bar{d} = \langle x_1, \ldots, x_n \rangle$, and outputs the predicted class label $\hat{y} \in \mathcal{Y}$. Specifically, we store a separate weight vector for each class and compute the sum of the weights of the unique n-grams present in each document. We include a bias term in the model. The class with the highest score is the predicted class of the document.

During training, the weight vectors $\mathbf{w}^c$ for $c \in \mathcal{Y}$ are randomly initialized such that $\mathbf{w}^c_i \sim \mathcal{N}(0, 1)$ for i in $1, .., |\mathcal{V}|$. We train the model for at most 100 epochs, and implement early-stopping if the model performance on the development set does not improve for 10 epochs consecutively.

Variables	Proper Name	Newsgroup
Hidden Layers	1	1
Num Neurons	10	15
Batch Size	300	128
Training Epochs	200	11
Optimizer	AdaGrad	Adam
Activation fcn.	sigmoid	sigmoid

Table 4: MLP Hyperparameter tuning

Model (Propername)	Accuracy
Development Results	
Perceptron (Comb., N=(1,4))	**0.8196**
Perceptron w/o Task-Specific Feat	0.7974
Perceptron w/o 4-Gram	0.7864
Perceptron w/o 3,4-Gram	0.6944
Perceptron w/o 1-Gram	0.2665
MLP (Comb., N=(1,4),sig.,AdaG.)	**0.8434**
MLP (tanh,AdaGrad)	0.8320
MLP (tanh,Adadelta)	0.8299
MLP w/o Task-Specific Feat	0.8154
MLP w/o 1-Gram	0.4901
Test Results	
Perceptron	0.8026
MLP	0.8424

Table 5: Performance on Propername. We evaluate the performance of the model when certain features are ablated. We mark in bold the best results.

MLP Implementation Details (8pt) In our implementation of the MLP, we make the number of the layers, the number of neurons in the layer, and the activation functions as variables, to make our model class flexible for tuning. We use the same features from the dataset described in Perceptron implementation as the input in the first layer and output class labels in the last layer with the softmax activation function. We use cross-entropy loss as the loss function. To optimize our model, we use grid-search method to find the best combination of the hyperparameters. We evaluate the model with its prediction accuracy with the development set. Our final hyperparameters for each MLP model are shown in Table 4.

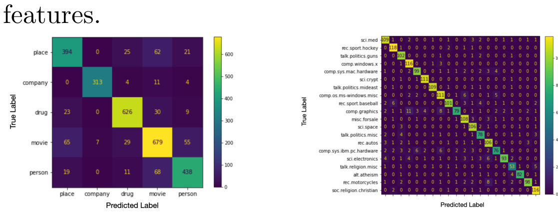
4 Results and Analysis

4.1 Proper Name Classification (12pt)

On the test set, we achieved an accuracy of 0.8026 using our Perceptron model, and the accuracy is improved to 0.8424 using our MLP model. The input features to both models include N-gram for $N \in 1, 2, 3, 4$ and the task-specific features.

Model (Newsgroup)	Accuracy
Development Results	
Perceptron (Comb., N=(1,3))	**0.8714**
Perceptron w/o Task-Specific Feat.	0.8608
Perceptron w/o 3-Gram	0.8612
Perceptron w/o 2,3-Gram	0.8608
Perceptron w/o 1-Gram	0.6195
MLP (Comb., N=(1,2), sigmoid,Adam)	**0.9152**
MLP (sigmoid,AdaGrad)	0.9089
MLP (tanh,Adam)	0.9046
MLP w/o 2-Gram	0.8984
MLP w/o 1-Gram	0.7548
Test Results	
Perceptron	0.7521
MLP	0.8111

Table 6: Performance on 20Newsgroup. We evaluate the performance of the model when certain features are ablated and when different activation functions and optimizers are used. We mark in bold the best results. *Comb. indicates combined features.



(a) Confusion matrix on Propername using MLP (Combined, N=(1,4), sigmoid, AdaGrad).

(b) Confusion matrix on 20newsgroup using MLP (Combined, N=(1,2), sigmoid, Adam).

Figure 1: Confusion Matrices

We performed feature and parameter search separately for each model. Our final Perceptron model is selected using the development set. We experiment with different combinations of N's for the N-gram feature, with and without the task-specific features. We observed a minor drop in performance when excluding the task-specific features and 4-Grams, but see bigger drops in accuracy when 1-Gram and 3-Gram are ablated, highlighting their importance in the classification task. We can also find the similar drop from the MLP model when 1-Gram are ablated.

From Figure 1a, we see that our MLP model is prone to error when a word may belong to either class under different context. For example, the name of a movie can also be the name of a 'person' (e.g. Queen Victoria) or of a 'place' (e.g. Peters). Such ambiguity makes it difficult for the model to distinguish these related classes.

4.2 Newsgroup Classification (12pt)

On the test set, we achieved an accuracy of 0.7521 using Perceptron, and the accuracy is improved to 0.8111 using MLP. The input features to both models include N-gram for $N \in 1, 2$ and the task-specific features. We also included 3-Gram for Perceptron.

We experiment with different combinations of N's for the N-gram feature, with and without the task-specific features. We observed a minor drop in performance when excluding the task-specific features, 2 and 3-Grams, but see bigger drops in accuracy when 1-Gram is ablated, high-lighting the importance of 1-Gram task with both Perceptron and MLP model.

From Figure 1b, we see the MLP model is prone to error when two classes of documents are related and share common key words. For example, the topic 'comp.graphics' is highly related to 'comp.windows.x', hence a lot of vocabularies and phrases are shared.

Batch Size	Average	S.D.
Proper Name		
1	333.17	2.81
300	17.06	0.06
500	16.56	0.44
1000	16.29	0.56
Newsgroup		
1	20.93	0.14
300	0.75	0.01
500	0.71	0.01
1000	1.35	0.53

Table 7: Batching Benchmarking

4.3 Batching Benchmarking (4pt)

The result of the batching benchmarking of MLP model on GPU is shown in Table 7. The speed is faster with batching than without. The reason is that using larger batch sizes would allow us to parallelize computations to a greater degree, hence speeding up the computation time. The training time of the Newsgroup is faster than the Proper Name in our result. The reason is that the epoch number of each model is different (11 for Proper Name and 200 for Newsgroup), so even the feature dimensionality of the Newsgroup is larger than the Proper Name, the training speed for Newsgroup's model is faster.

5 Conclusion (4pt)

Our Perceptron and MLP models are fairly generalizable and achieve reasonable accuracy on the 2 classification tasks. We found that 1-Gram is important for the performance in both datasets, and MLP consistently outperforms Perceptron. Qualitative analysis of our models show that most errors are made when distinguishing between ambiguous names and between related topics with the same theme.