Linguistic Alignment : Classifying Social Power Status using Neural Networks

ABSTRACT

The world around us has changed drastically by the introduction of internet in the year 1969. The 21st century has seen a lot of online communities formed where a lot of information is exchanged. To understand these online communities, it is of utmost importance to understand the social interaction within these communities. Over the years, studies have shown that social power plays a role in linguistic alignment between dialogue partners.

Using the above assumption, we have conducted experiments on two different datasets with the implementation of convolutional and recurrent neural networks models on input of shallow utterance features such as part-of-speech (POS) tags and Linguistic Inquiry and Word Count (LIWC) markers to show that a classification-based model can be used to predict the relative social power statuses of the speakers.

1. Introduction

Social networking services are now a part every person's life. People often use these services to stay in-touch with other people, talk about an on-going issue, share knowledge to others and much more. Tons of data are generated everyday through these services. It would be useful if we could extract relevant information from these kinds of data.

Linguistic alignment is a behaviour where one speaker tends to adapt grammatically based on the utterance of another speaker. A lot of research has been going on for years in the field of linguistic alignment with various computational models being proposed which include the probability based (Danescu-Niculescu-Mizil et al., 2012), repetition-based (Fusaroli et al., 2012; Wang et al., 2014), inference-based (Doyle and Frank, 2016), generalized linear models (GLMs) (Reitter and Moore, 2014; Xu et al., 2018) etc.

The findings of probability based computational model (Danescu-Niculescu-Mizil et al., 2012) shows that the people with higher status are bound to receive more alignment than those of lower power. With the presumption of the probability based computational model, we propose a neural network-based model for linguistic alignment which can predict the relative social power status of the pair of utterance. This model is suitable for a dataset comprising of utterances from two relative status of social power. Neural networks have proven to be great models in classification problems and would be useful in classifying whether a pair of utterance belongs to a H◄─L or L◄─ H category.

2. Defining the problem

Based on findings of Danescu-Niculescu-Mizil et al., 2012, we know that linguistic alignment is affected by relative social power statuses of the speaker. The idea here is to build computational models that can help predict the relative social power status on a pair of utterances. We make use of neural networks which is a supervised learning, since neural networks have many natural language features available to tackle such problems. Furthermore, we can also make use of POS (Part-of-speech) tags as well as LIWC tags to provide inputs to our models.

The datasets used comprise of conversations between a high-status speaker and a low-status speaker (Example: Judge and Lawyer). There are 4 different types of classes for the pair of utterances which are H ◄─ H (conversation between high-status speaker and high-status speaker), L◄─ L (conversation between low-status speaker and low-status speaker), H ◄─ L (conversation between high-status speaker and low-status speaker) and L◄─ H (conversation between low-status speaker and high-status speaker).

We now need to find a model which can best classify the pair of utterance in one of the 4 above categories.

3. Data

We make use of two types of datasets:

Supreme court dataset. The dataset contains sentences from judge(JUSTICE) and

lawyer(NOT-JUSTICE). Here the judge serves as high-status speaker and lawyer serves as low status speaker. The dataset contains 51498 statements.

Wikipedia dataset. This dataset comprises of the conversations between Wikipedia admins and Wikipedia users. The Wikipedia admins act as the high-status speaker and Wikipedia users act as the low status speaker. There are 516766 statements within the dataset.

4. Data Pre-Processing

Creating pair of utterances

Using the initial datasets, we create a new dataset consisting of utterance pairs. In this new dataset, each pair contains 2 sequences, one from high-status speaker and the other from the low-status speaker. We assign labels H◄— L and L◄—H depending on the position of the sequence. If the line consists of high-status speaker sentence followed by low-status speaker sentence, we assign the label H◄—L, else if the line consists of low-status speaker sentence followed by high-status speaker sentence we assign the label L◄—H. Remaining all the lines are discarded (ones containing high-status sentence followed by high-status sentence and low-status sentence followed by low-status sentence).

Below is an example of a conversation between a teacher and a student.

Teacher : What are you working on?
Student : I am trying to solve this mathematical problem.
Teacher : Do you need any help with this?
Student : No, I will be able to do it. Thanks.

The sample utterances pairs would like:

Teacher : What are you working on? Student : I am trying to solve this mathematical problem.	H◄—L
Student : I am trying to solve this mathematical problem. Teacher : Do you need any help with this?	L◄—H
Teacher : Do you need any help with this? Student : No, I will be able to do it. Thanks.	H◄—L

Converting words into POS and LIWC tags

We convert these utterances in the sentences containing POS-Tags. The sentence “yes and the same answer for 1981” would be converted to “INTJ CCONJ DET ADJ NOUN ADP NUM”.

Below table describes the different types of part-of-speech:

Part-Of-Speech	
SPACE	DET
X	NOUN
SYM	VERB
CCONJ	ADV
PRON	NUM
PUNCT	PART
ADJ	ADP
PROPN	INTJ

Part-of-speech tagging (POS tagging), also called grammatical tagging or word-category disambiguation, is the process of marking up a work in a text as corresponding to a particular part of speech.

We also convert the utterances into sentences containing LIWC tags. The sentence “if you have uploaded other non free” would be converted into “excl discrep conj tentat ppron auxverb x”.

LIWC is a computerized text analysis program that outputs the percentage of words in a given text that fall into one or more of over 80 linguistic, psychological, and tropical categories.

Creating bag-of-words

The bag-of-words model is a simplifying representation used in natural language processing. The bag-of-words is an essential part for our models. Every word that occurs once in the dataset is used to build the bag-of-words. From the sample conversation of teacher and student discussed earlier the bag-of-words would look like:

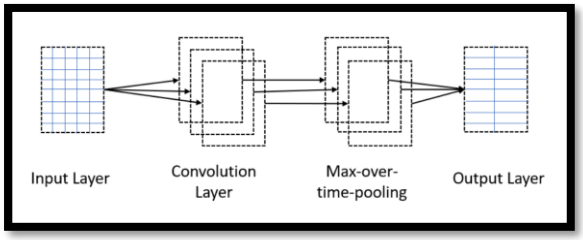
Bag-of-words		
What	Are	You
Working	On	I
Am	Trying	To
Solve	This	Mathematical
Problem	Do	Need
Any	Help	With
This	No,	Will
Be	Able	Thanks

It can be seen that the words “I”, “You”, “This”, “To” and “Do” repeat multiple times but we use them only once in our bag-of-words.

5. Model Implementation

CNN Model

The CNN model is designed to use word embedding and one-hot encoding. The word embedding takes the sentences as input for the model architecture with 2 channels. Whereas, the one-hot encoding uses the part-of-speech tags for each sentence as input for the model architecture with two channels. The below image describes the model pictorially.



Input Layer: The input layer takes a batch of sentences from the part-of-speech tags for each sentence, which is then converted into numeric values using word2id representation. Each batch is of size batch-size x max-length, where, max-length is the number of the words present in the longest sentence present in the batch. This input is padded with zeros for missing values i.e. sentences with shorter lengths than that of max-length.

Convolution Layer: The convolution layer is used to perform the convolution operation on the embedding matrix, with a kernel of varying height but constant width which is same as the embed_size. After performing the convolution operation, we apply the Relu activation followed by max_pool operation on each tensor for each kernel height that we provide to the model. We then filter the maximum activation for each channel and concatenate the resulting tensors. The output of the concatenated tensors is fully connected to the output layers comprising of two units.

Output Layer: The output layer is a fully connected SoftMax layer whose output is the probability distribution over the labels. The size of this layer is batch-size x output size, where, the output size is the number of possible outcomes for the model.

RCNN Model

Input Layer: The input layer takes a batch of part-of-speech tags for each sentence, which is then converted into numeric values using word2id representation. Each batch is of size batch-size x max-length, where, max-length is the number of the words present in the longest sentence present in the batch. This input is padded with zeros for missing values i.e. sentences with shorter lengths than that of max-length.

Convolution Layer: The word of each sentence is combined with a context, to help get a more precise word meaning. In implementing this model, we make use of a recurrent structure to help obtain the contexts. Let us consider a word w_i using which we define the left context of the word as $c_l(w_i)$ and the right context of the word as $c_r(w_i)$. Below definitions are used to define $c_l(w_i)$ and $c_r(w_i)$:

$$c_l(w_i) = f(W^{(l)}c_{l(w_{i+1})} + W^{(sl)}e_{(w_{i-1})})$$

$$c_r(w_i) = f(W^{(l)}c_{r(w_{i+1})} + W^{(sr)}e_{(w_{i-1})})$$

Where $W^{(l)}$ is a matrix used to transform the hidden layer into the next hidden layer and $W^{(sl)}$ is a matrix used to combine the next word's left context with the semantic of the current word. The word embedding of the word is defined as $e(w_{i-1})$.

The equation below is used to concatenate the left-side context word $c_l(w_i)$, the word embedding $e(w_i)$ and the right-side context word $c_r(w_i)$ to produce the representation of word w_i.

$$x_i = [c_l(w_i); e(w_i); c_r(w_i)]$$

We now perform a linear transformation on the word w_i representation in x_i using tanh activation function and pass this result to the next layer in the model.

$$y_i^{(2)} = \tanh(W^2 x_i + b^2)$$

Once the word representations are calculated, we apply the max-pooling layer which attempts to find the most import semantic factors.

$$y^{(3)} = \max_{i=1} n\ (y_i^{(2)})$$

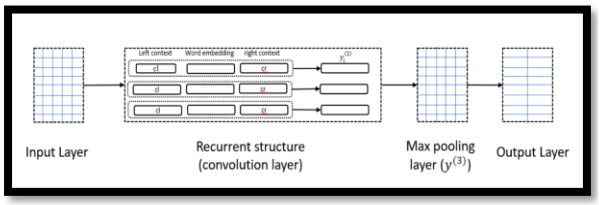
For the final step in our model, we apply a linear transformation like the traditional neural networks as given in the equation below:

$$y^{(4)} = W^{(4)}y^{(3)} + b^{(4)}$$

The model takes in the sentences from the input layer on which embedding is performed. We use this embedding matrix to perform an LSTM operation followed by concatenation of the LSTM output which gives us the word representation. We perform a linear transformation the word representation using the tanh activation function and pass the result to the next layer. In the next layer, we perform a max pooling which attempts to find the most import semantic factors. For the final step in our model, we apply a linear transformation like the traditional neural networks and connect this layer to the softmax layer to produce an output layer comprising of two units.

Output Layer: The output layer is a fully connected SoftMax layer whose output is the probability distribution over the labels. The size of this layer is batch-size x output size, where, the output size is the number of possible outcomes for the model.

Below is the pictorial representation of RCNN model:

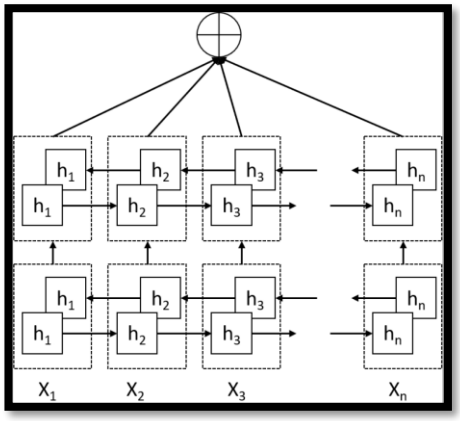


LSTM Model

To take text as the input of Neural Networks, one-hot and word-embedding methods were introduced to this model. When using one-hot method, every word of every sentence were instead by its part of speech. So, thousands of words were replaced by 15 POS tags. For word-embedding, every word was presented as a 256-dimensional vector based on their meaning. LSTM layer is introduced in this model because of its high performance in sequence-based tasks. A fully connected layer with softmax activation function is linked to the LSTM layer to output binary data.

LSTM Attention Model

LSTM-Attention model is similar to the LSTM based model. The only difference between LSTM-Attention model and LSTM based model is that attention layer is added into the LSTM model. All of the hidden states from h_0 to h_t are concatenated together. For each word in the sentence, we annotate them with an attention weight. After multiplying the attention weight and hidden states, the softmax activation function is used to normalize attention scores. A fully-connected layer is linked to the attention score to generate the output.



6. Running the models and their results

The 4 models i.e. CNN, RCNN, LSTM and LSTM attention were run on 3 different types of datasets for both Wikipedia and Supreme court datasets. We define the 3 types of datasets as follows:

Embedding : The embedding data contains the sentences from the initial datasets but arranged in pairs of sequences.

POS : The pair of sequences generated in embedding dataset is converted into POS tags.

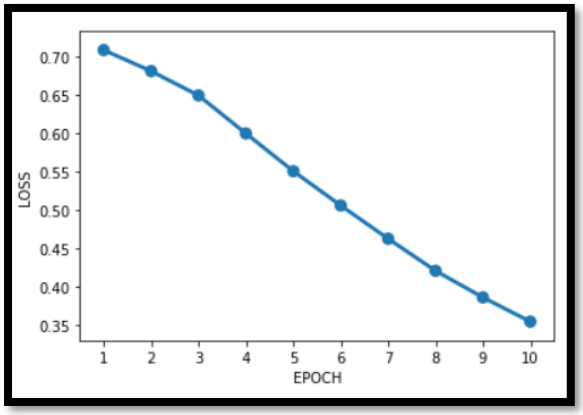
LIWC : The pair of sequences generated in embedding dataset is converted into LIWC tags.

Throughout this paper we have used the data for H◄— L and L◄— H classes for classification, unless specified with different class.

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	65.48	62.20	58.31
	RCNN	76.15	67.05	58.17
	LSTM	63.22	63.85	58.36
	LSTM Attention	72.09	64.04	58.11
Supreme Court Dataset	CNN	68.73	64.42	61.00
	RCNN	86.36	76.00	76.03
	LSTM	76.11	74.36	72.81
	LSTM Attention	68.65	72.31	71.45

Results after running the models on the dataset

The loss functions are helpful in understanding if the model performance increases or decreases. From the below figure, we observe that at each epoch the loss function keeps reducing indicating that the models have been performing as per our needs. Similar trends have been observed in the remaining 3 models as well.



EPOCH vs LOSS plot for CNN model

The dataset was split into training and testing with 95% and 5% as the ratio respectively. The results shown through out this paper correspond to the test accuracy generated by each model indicating how well the models were able to classify the pair of utterances into one of the classes.

7. Fake Dataset Experiment

What if we change the sequence order of pair of utterances in our data? The problem that we are working with is a classical classification problem. In that case, the order of pair of utterances should not impact the results we obtained earlier. Since we are working with pairs of H◄— L and L◄— H, we can have 3 scenarios here:

Shuffle the order of all pair of utterances

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	64.57	61.95	58.34
	RCNN	75.57	66.08	58.80
	LSTM	62.55	64.40	58.83
	LSTM Attention	72.53	64.04	58.64
Supreme Court Dataset	CNN	66.60	62.67	60.07
	RCNN	85.35	75.88	75.61
	LSTM	74.40	72.50	71.53
	LSTM Attention	79.22	73.43	70.60

Shuffle the order of only L◄— H pair of utterances

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	66.25	61.73	58.56
	RCNN	74.88	64.81	59.27
	LSTM	61.95	63.33	58.17
	LSTM Attention	71.27	62.33	58.45
Supreme Court Dataset	CNN	68.00	64.66	62.36
	RCNN	85.78	76.19	75.37
	LSTM	75.33	72.85	71.84
	LSTM Attention	79.88	72.03	71.18

Shuffle the order of only H◄— L pair of utterances

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	64.70	61.75	58.11
	RCNN	75.82	67.10	58.69
	LSTM	62.06	63.13	58.91
	LSTM Attention	72.34	63.19	58.25
	CNN	68.54	65.47	61.32

Supreme Court Dataset	RCNN	85.32	77.00	76.38
	LSTM	75.88	72.31	72.73
	LSTM Attention	80.07	73.70	71.84

From the results above, we can infer that the performance doesn't degrade on shuffling the data. Hence, the order of pair of utterances doesn't matter in our case.

8. Shuffled Words in the Dataset Experiment

In this scenario, we shuffle the words within each pair of utterances to see if the performance of our models remains the same.

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	58.72	58.31	58.06
	RCNN	58.25	58.25	58.09
	LSTM	57.87	57.98	57.89
	LSTM Attention	58.11	57.87	57.87
Supreme Court Dataset	CNN	52.46	53.94	52.50
	RCNN	52.38	52.97	53.32
	LSTM	50.33	50.71	50.83
	LSTM Attention	50.67	50.60	50.71

The results above indicate the performance of our models drop. The reason for this behaviour is that with words shuffled in each pair of utterance the model is unable to distinguish precisely whether the given sentence is a H ◄— L pair or a L◄—H pair.

9. Pre-defined Bag-of-words Experiment

We have been using our own build bag-of-words until this point. We now use a predefined bag-of-words model called glove. The glove bag-of-words contains about 400,000 words in its predefined model. Below are the results of the models after running them with glove bag-of-words.

Dataset	Model	Embedding
Wikipedia Dataset	CNN	58.22
	RCNN	64.51
	LSTM	56.38
	LSTM Attention	62.73
Supreme Court Dataset	CNN	64.11
	RCNN	83.10
	LSTM	70.32
	LSTM Attention	63.97

The accuracy of the models when run with glove dips a little when compared to accuracy of the models run with our own build of bag-of-words. There are a lot more words present in the glove as compared to our own build bag-of-words which results in a slight dip in accuracy. Yet, the models show promising results when used with a predefined bag-of-words.

10. More Experimentations

So far, our data for Wikipedia and Supreme Court consisted of data involving **High-Low** and **Low-High** sentences. In order to explore more on our dataset, we worked on 3 new combinations on Wikipedia dataset:

H◄— * and L ◄— *

We use two new labels to run our models. We use pair of utterances which are H ◄— * (H◄— L and H◄— H) and L◄— * (L◄— L and L◄— H).

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	70.78	68.88	63.16
	RCNN	72.82	68.79	62.42
	LSTM	68.57	68.72	68.63
	LSTM Attention	71.20	68.65	68.57

***◄—H and *◄—L**

We use two new labels to run our models. We use pair of utterances which are * ◄— H (L◄— H and H◄— H) and * ◄— L (L◄— L and H◄— L).

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	67.36	63.34	62.87
	RCNN	68.33	62.54	62.49
	LSTM	62.40	62.30	62.28
	LSTM Attention	68.21	62.46	62.41

H ◄—H and L◄—L

We use two new labels to run our models. We use pair of utterances which are H ◄— H and L◄— L.

Dataset	Model	Embedding	POS	LIWC
Wikipedia Dataset	CNN	87.27	78.35	78.62
	RCNN	87.53	78.28	78.19
	LSTM	78.14	78.14	78.21
	LSTM Attention	86.11	78.16	78.21

The models can predict the relative social power status in all the above scenarios with good results. The case H ◄— H and L◄— L shows best performance which goes on to show how well the models can distinguish conversations between the speakers with same or different social power statuses.

11. Conclusion

This work shows that it possible to have models for linguistic alignment which can help predict the relative social power status of a pair of utterance using neural network with promising results. The models were trained under different experiments in order to show the sustainability of these models and they didn't fail to do so.

We were also able to learn that using part-of-speech (POS) tags and linguistic inquiry and word count (LIWC) tags in place words can be useful in classifying datasets where we have utterances from two relative statues of social power.

12. Acknowledgement

I sincerely thank Yang Yu for his guidance and advice throughout this work.

13. References

Cristian Danescu-Niculescu-Mizil,LillianLee, Bo Pang, and Jon Kleinberg. 2012. Echoes of power: Language effects and power differences in social interaction. In Proceedings of the 21st International Conference on World Wide Web, pages 699–708, Lyon, France. ACM.

Gabriel Doyle and Michael C Frank. 2016. Investi-gating the sources of linguistic alignment in conversation. In Proceedings of the 54th Annual Meet- ing of the Association for Computational Linguistics, pages 526–536, Berlin, Germany. Association for Computational Linguistics.

Riccardo Fusaroli, Bahador Bahrami, Karsten Olsen, Andreas Roepstorff, Geraint Rees, Chris Frith, and Kristian Tylen. 2012. Coming to terms quantifying the benefits of linguistic coordination. Psychological Science, 23(8):931–939.

James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count: LIWC 2001. Mahway: Lawrence Erlbaum Associates, 71:2001.

Martin J Pickering and Simon Garrod. 2004. Toward a mechanistic psychology of dialogue. Behavioral and Brain Sciences, 27(02):169–190.

David Reitter and Johanna D Moore. 2014. Alignment and task success in spoken dialogue. Journal of Memory and Language, 76:29–46.

Yafei Wang, David Reitter, and John Yen. 2014. Linguistic adaptation in online conversation threads: analyzing alignment in online health communities. In Proceedings of the Fifth Workshop on Cognitive Modeling and Computational Linguistics (at ACL), pages 55–62, Baltimore, Maryland. ACL.

Yang Xu, Jeremy Cole, and David Reitter. 2018. Not that much power: Linguistic alignment is influenced more by low-level linguistic features rather than social power. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistic (Volume 1: Long Papers), volume 1, pages 601–610.

Yang Xu, Jeremy Cole, and David Reitter. 2019. Linguistic alignment is affected more by lexical surprisal rather than social power. Proceedings of the Society for Computation in Linguistics, 2(1):349–352.