Hate Speech Detection Using Machine Learning

Hiroki Endo

Task

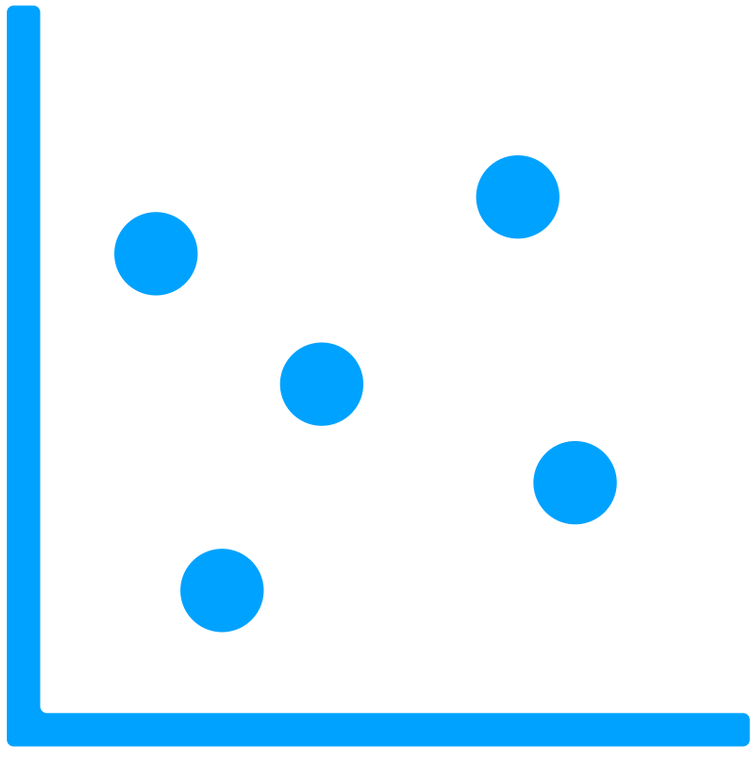
Train machine learning model to classify tweets from Twitter to following 3 classes

hate speech

offensive language

neither

Data Overview



Data Overview

Sharkevin RIP Sean P
@WheresMyJuice

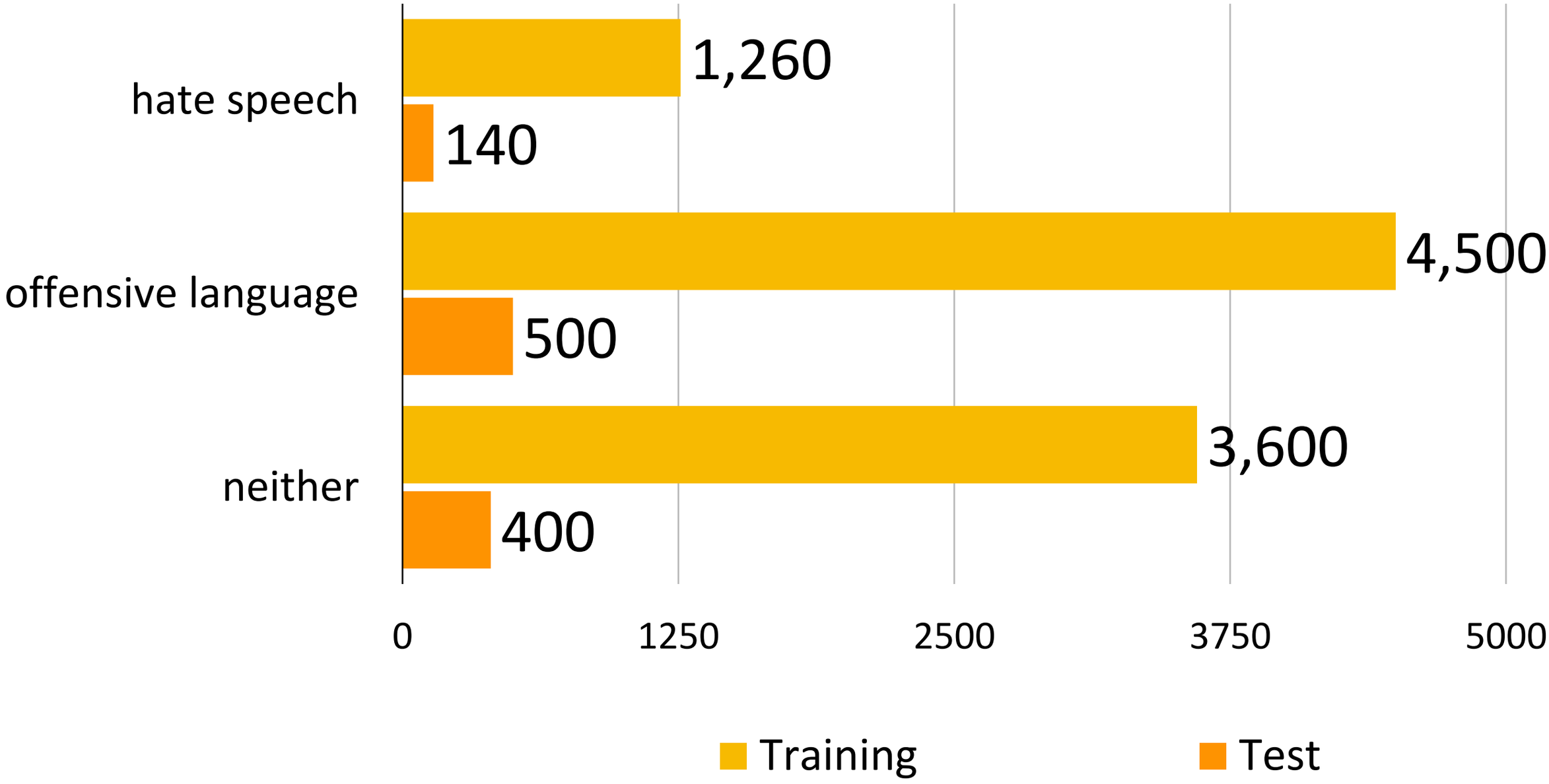
cookies and crackers aren't even on the same level!

午前7:04 · 2014年6月27日 · Twitter Web Client

↓

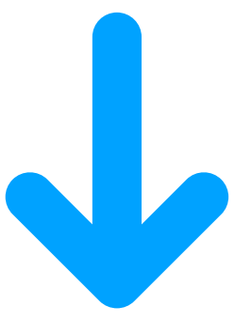
cookies and crackers aren't even on the same level!

Data size by class



Data Augmentation with Google Translate

Original:
Fuck you pussy ass hater go suck a dick and die fast



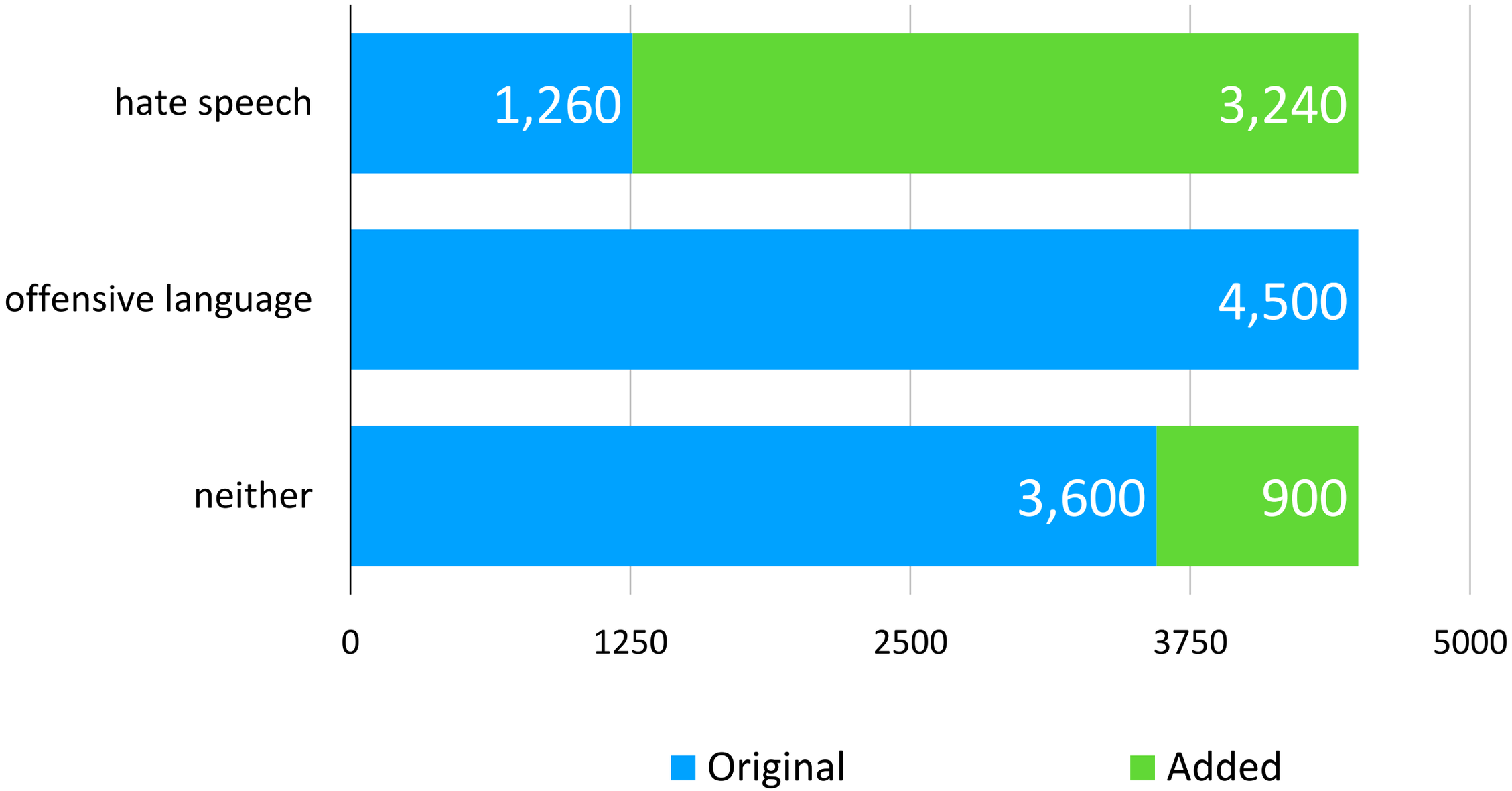
Translate to French, German, Dutch

Translated to French:
Va te faire foutre le cul chatte aller sucer une bite et mourir vite

Translate back to English

After re-translation:
Fuck you pussy hater will suck a dick and die quick

Data After Augmentation[8]



Preprocessing: Replacement

"@John: How was Japan?" Awesome！ 😀
RT@Hana miss Japan : (（😀）

@ mention to <user>

Indicate Reply

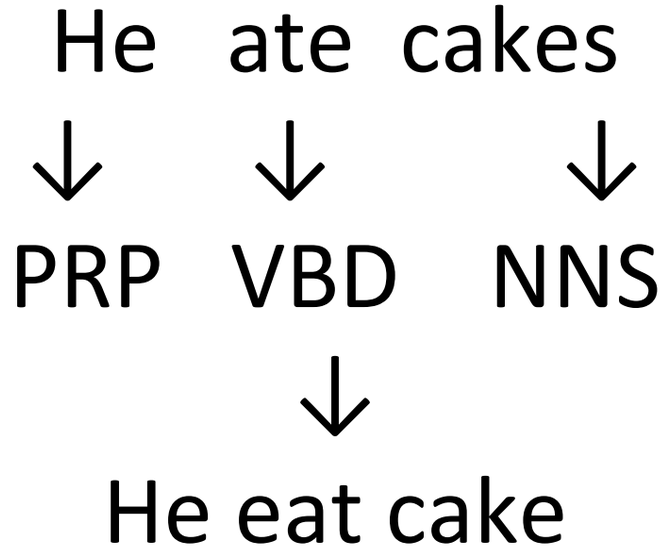
"<user>: How was Japan?" <reply> Awesome! : grinning_face:
<rt_from> miss Japan : sad_face:

Indicate Retweet

Replace emoji with corresponding meaning

Preprocessing: lemmatization

- Used Stanford CoreNLP Package[1] → Pipeline for doing various preprocessing



- POS tagging using GATE Twitter POStagger[2]

→ POS tagger trained with twitter tweet

- Lemmatize based on resulting POS tags

1. https: //stanfordnlp.github.io/CoreNLP/extensions.html
2. https: //gate.ac.uk/wiki/twitter-postagger.html

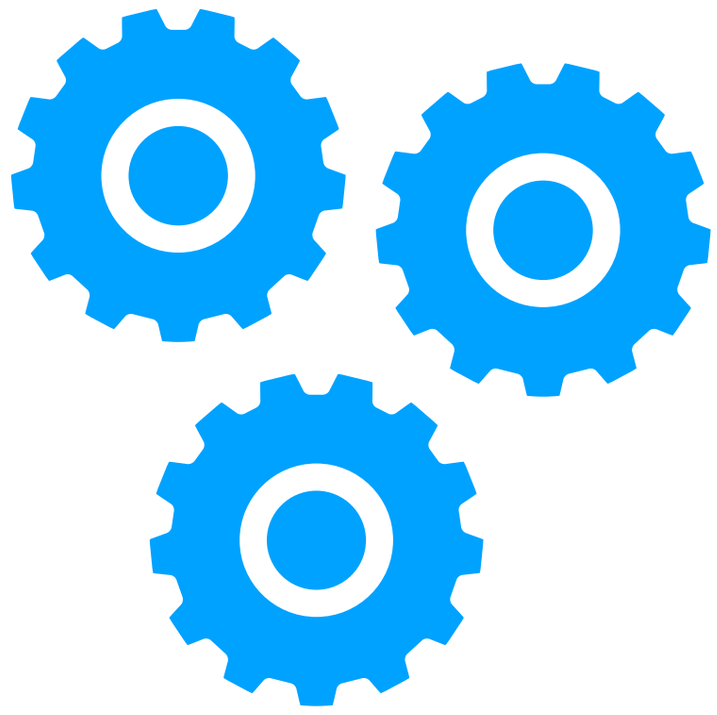
Preprocessing: Others

- **Spell correct**

→ Used module autocorrect[1]

- **Fix abbreviation**

→ Used module contractions[2]



- **Remove punctuations other than ?!**
- **Make Everything into lower case**
- **Remove stopwords**

→ Used built in list from nltk[3]

1. https: //pypi.org/project/autocorrect/
2. https: //pypi.org/project/contractions/
3. https: //www.nltk.org/

Chosen Models

- Decision Tree
- LSTM

Decision Tree Parameter

	Only Original Data	Augmented Data
Max Depth	20	30
Minimum sample to create new branch	17.5%	15.5%
Feature considered at each branch	84%	82%

LSTM
Parameters ▪ Model Structure

	Only Original Data	Augmented Data
embedding	20	20
LSTM Blocks	91	171
dropout	47%	63%
Softmax	3	3
Learning Rate	0.001	0.001

Model Structure

1. embedding layer

2. LSTM layer
Optimizer: Adam
Activation Func: layer → tanh
gates → hard sigmoid

3. Dropout

4. Output layer: softmax

Feature Extraction

- Decision Tree: TF-IDF

  A. Give bigger weight to words appearing frequently in a text, and less frequently across the whole data.

  B. Words, which occurrences are in top 30%, and bottom 0.4%, are ignored

  C. Combination of unigram and bigram

Feature Extraction

- LSTM : let model learn embedding

→allows to create embedding best fitted to the task

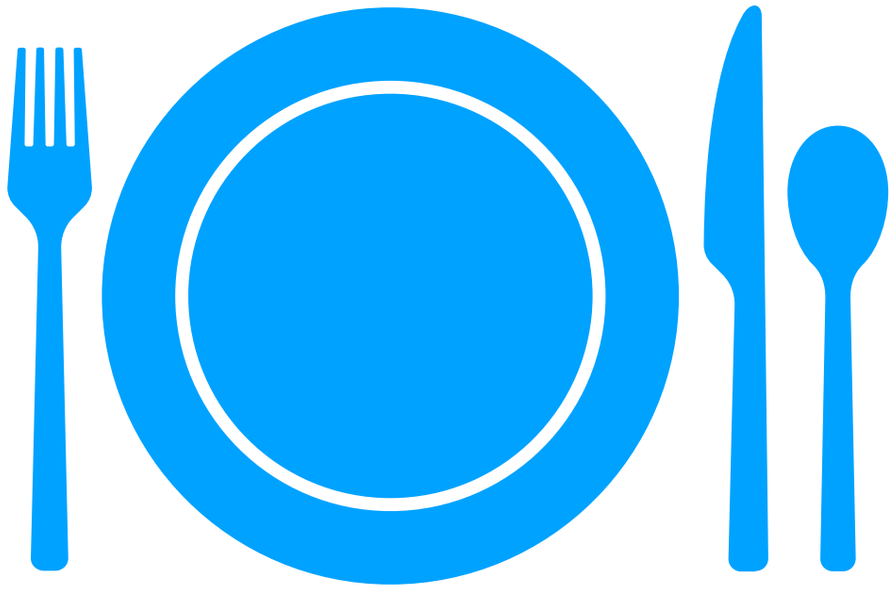
ex. words with closest cosine distance to 'white'

GloVe[1] (Twitter base）: black, blue, green, yellow, red, purple, tank

Embedding created by model : nigga, fuck, faggot, nigger, fag, ass, racist

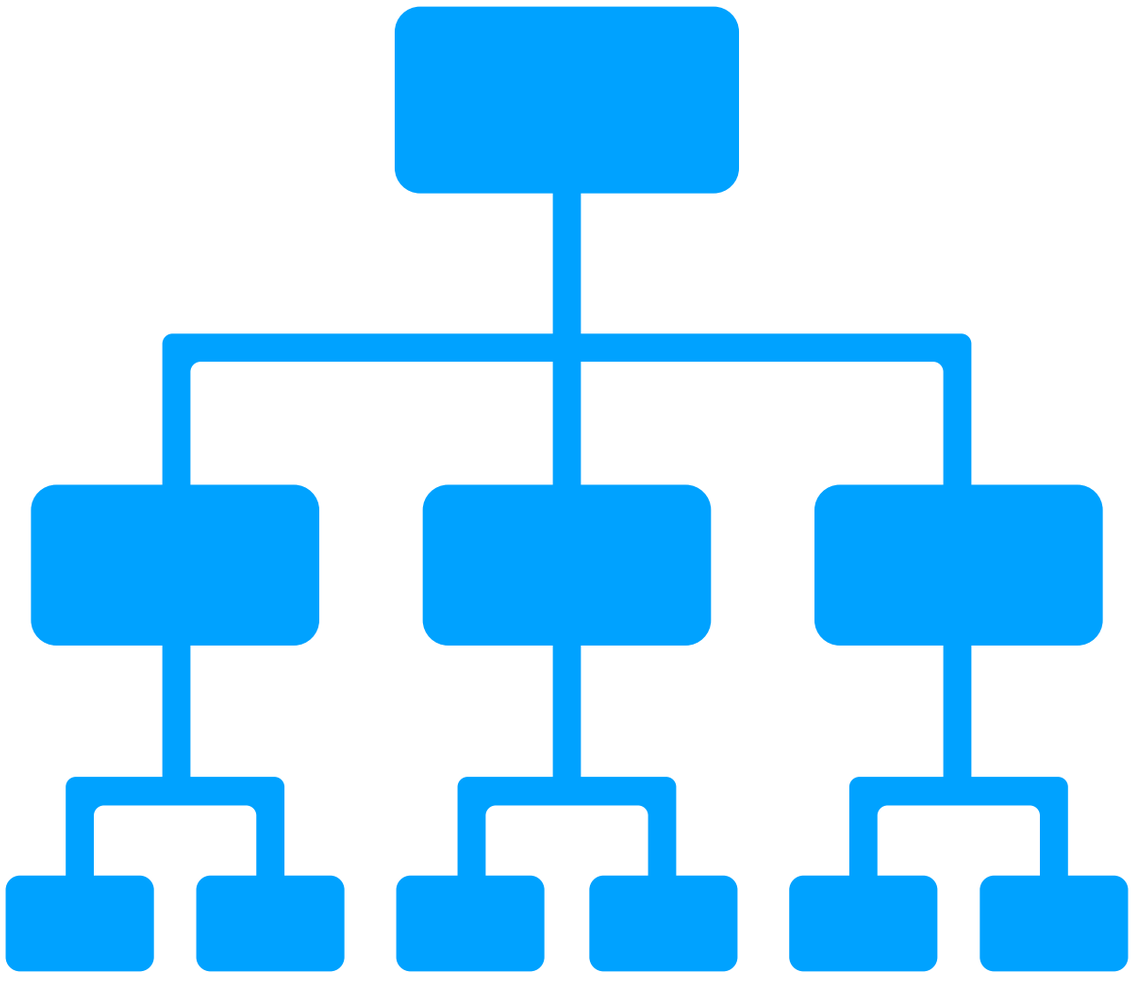
1. https: //nlp.stanford.edu/projects/glove/

Result

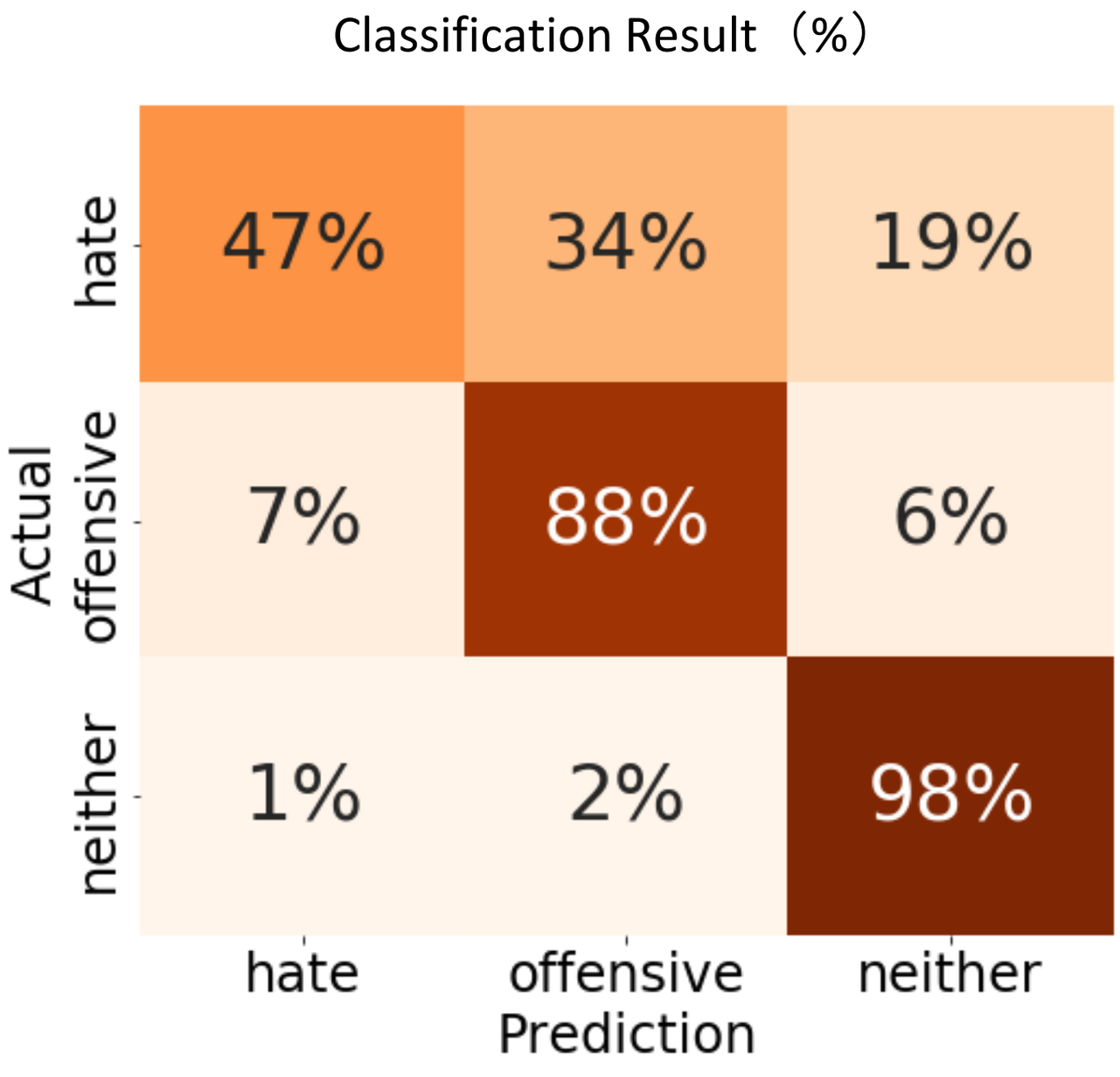


Decision Tree

CART



Result: Original Data Only



- Model can detect offensive language, neither well

  f1-score 90%

- Not successful at detecting hate speech

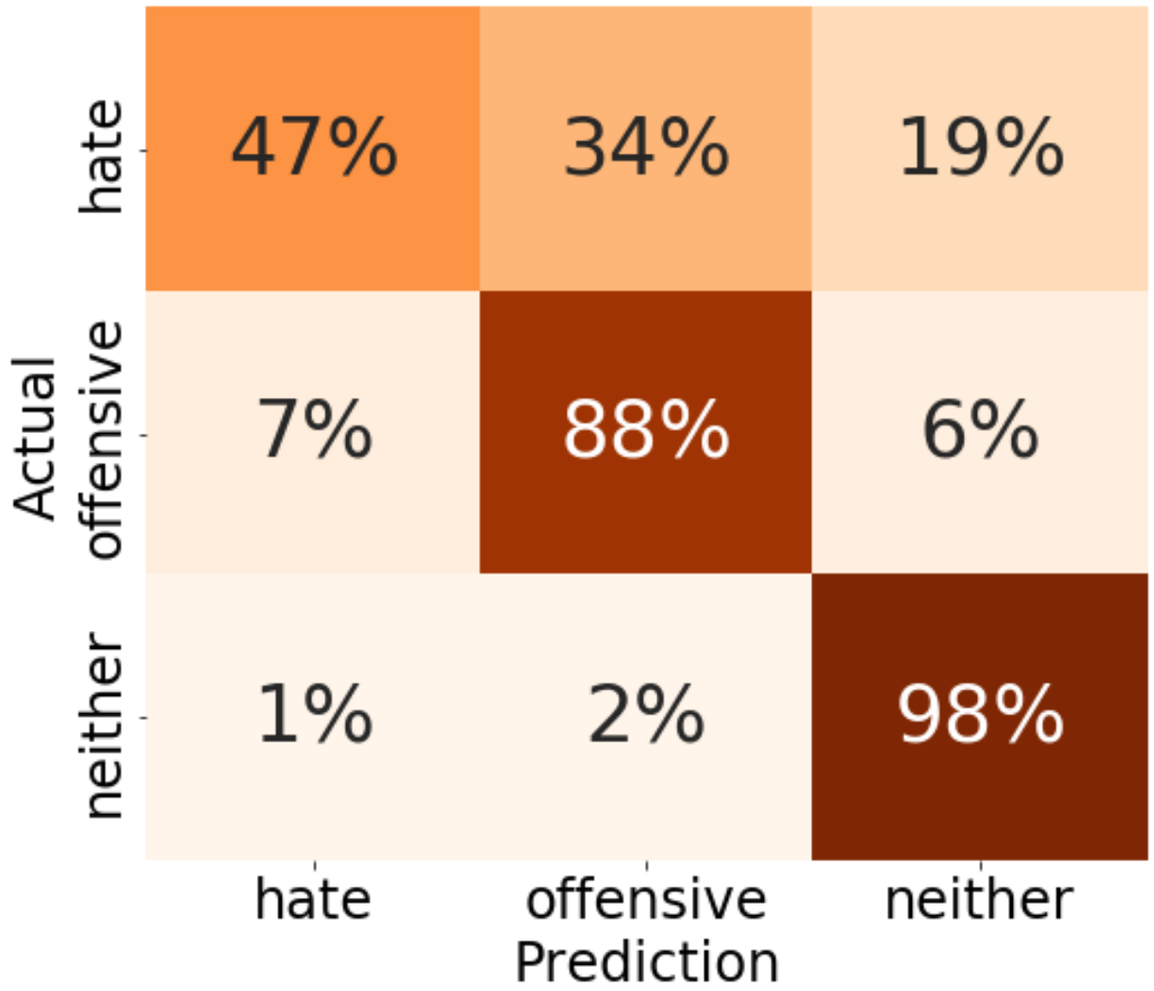
- Especially, there is problem distinguishing hate speech and offensive language

- f1-macro: 78%

- f1-weighted: 85%

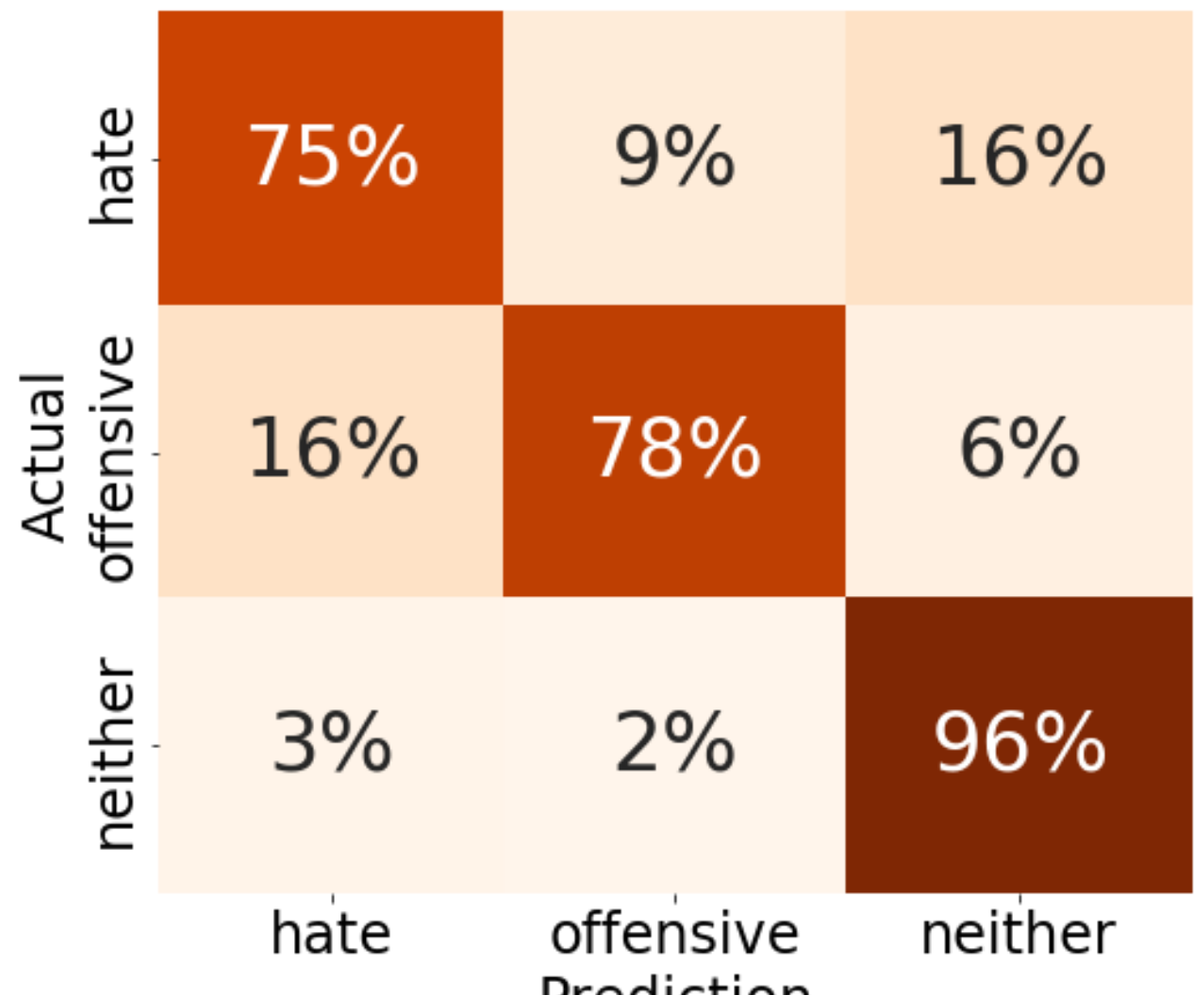
Result: Decision Tree

Without Data Augmentation



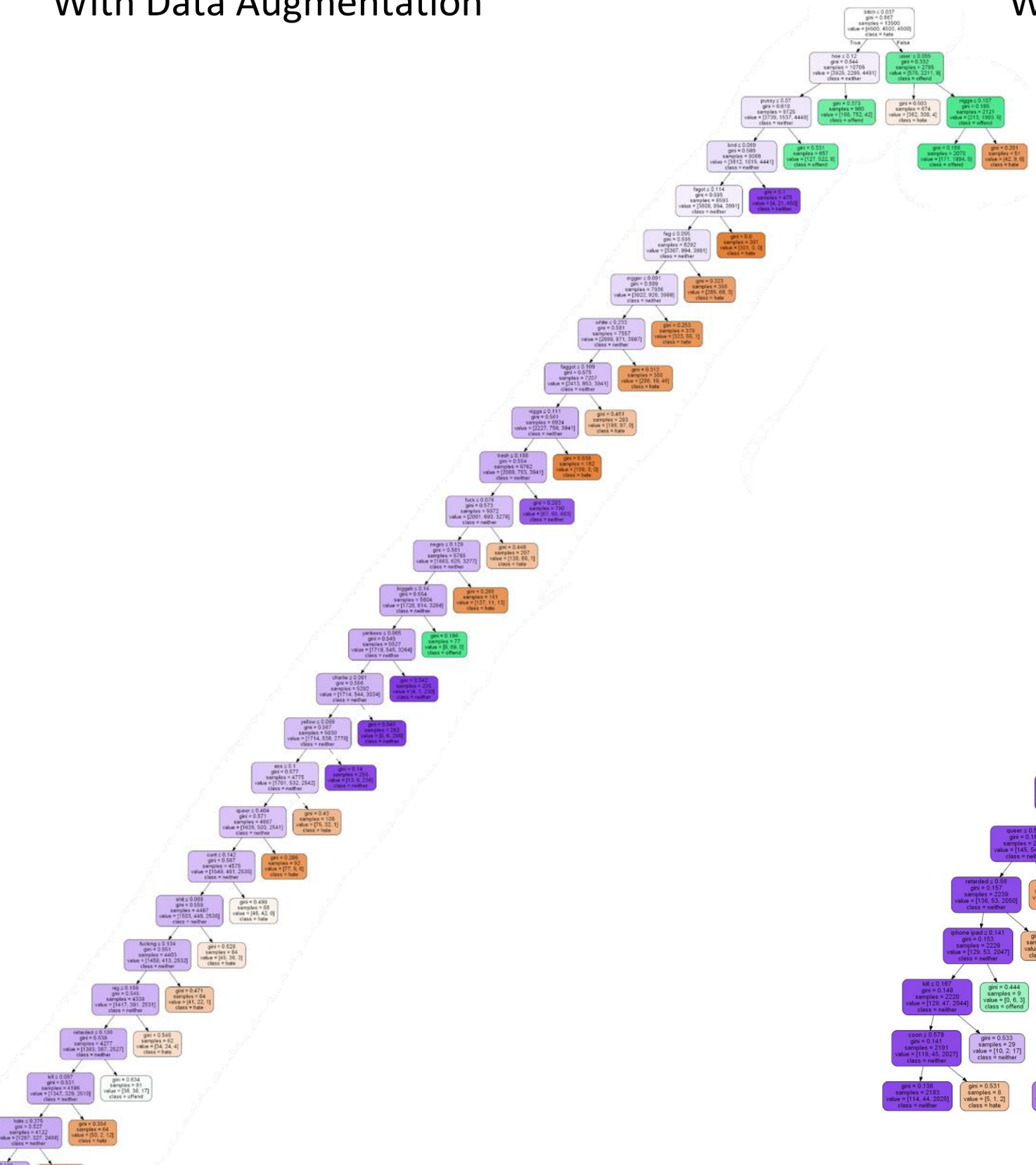
- hate speech's true positive improved greatly
- More data was mistakenly classified as hate speech

With Data Augmentation

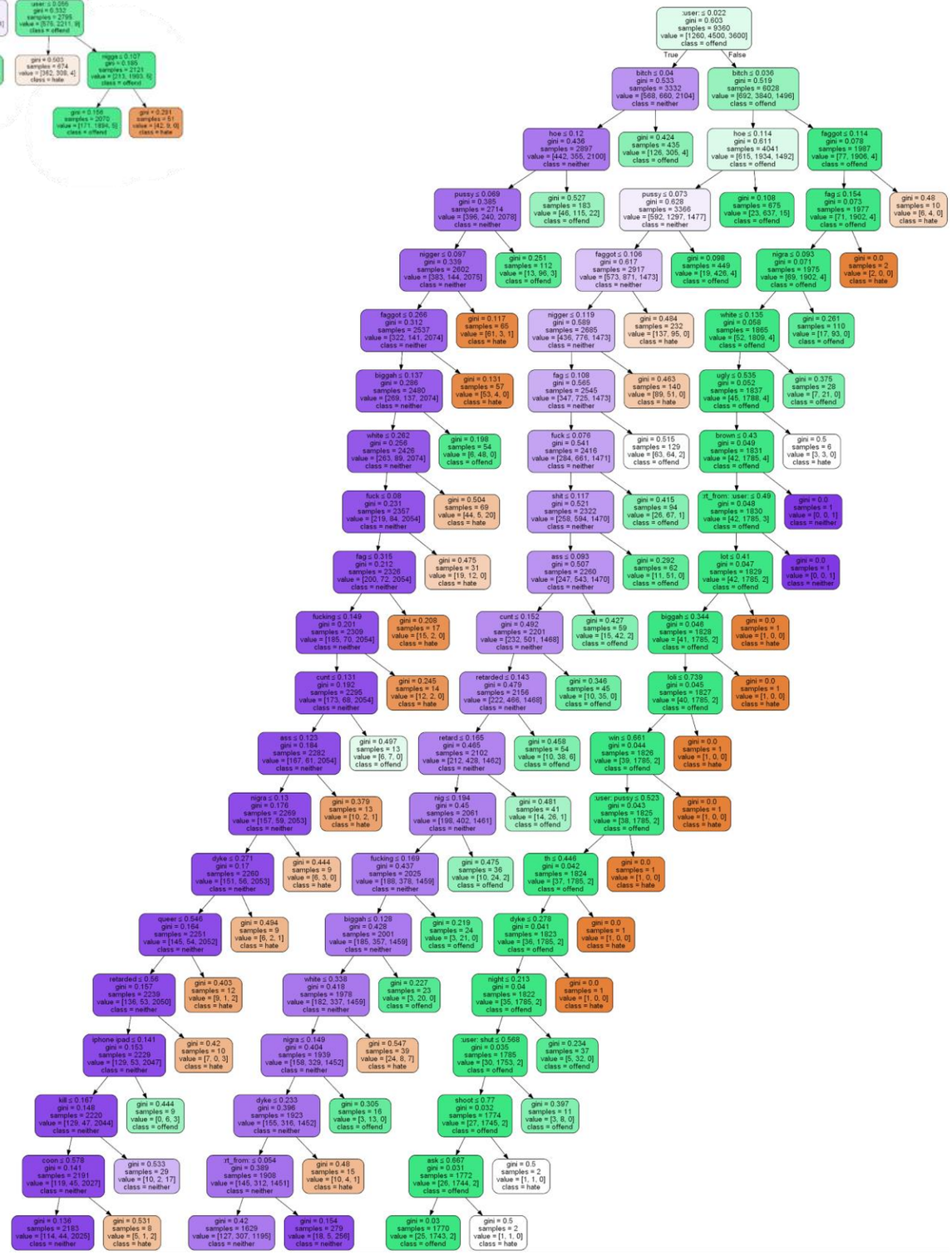


- f1-macro: 78% → 80%
- f1-weighted: 85% → 85%

With Data Augmentation



Without Data Augmentation



Contains ‘bitch’ → most likely not in neither class

bitch ≤ 0.037
gini = 0.667
samples = 13500
value = [4500, 4500, 4500]
class = hate

Bitch

value =
[hate, offend, neither]

True

False

hoe ≤ 0.12
gini = 0.644
samples = 10705
value = [3925, 2289, 4491]
class = neither

:user: ≤ 0.055
gini = 0.332
samples = 2795
value = [575, 2211, 9]
class = offend

575, 2211, 9

Nigga

pussy ≤ 0.07
gini = 0.618

gini = 0.373

gini = 0.503
samples = 674
value = [362, 308, 4]
class = hate

nigga ≤ 0.107
gini = 0.185
samples = 2121
value = [213, 1903, 5]
class = offend

In some case, text is classified as neither just by having ‘bird’ in it.

Bird

bird ≤ 0.069
gini = 0.589
samples = 9068
value = [3612, 1015, 4441]
class = neither

gini = 0.331
samples = 657
value = [127, 522, 8]
class = offend

gini = 0.156
samples = 2070
value = [171, 1894, 5]
class = offend

gini = 0.291
samples = 51
value = [42, 9, 0]
class = hate

Offensive Class

Hate Class

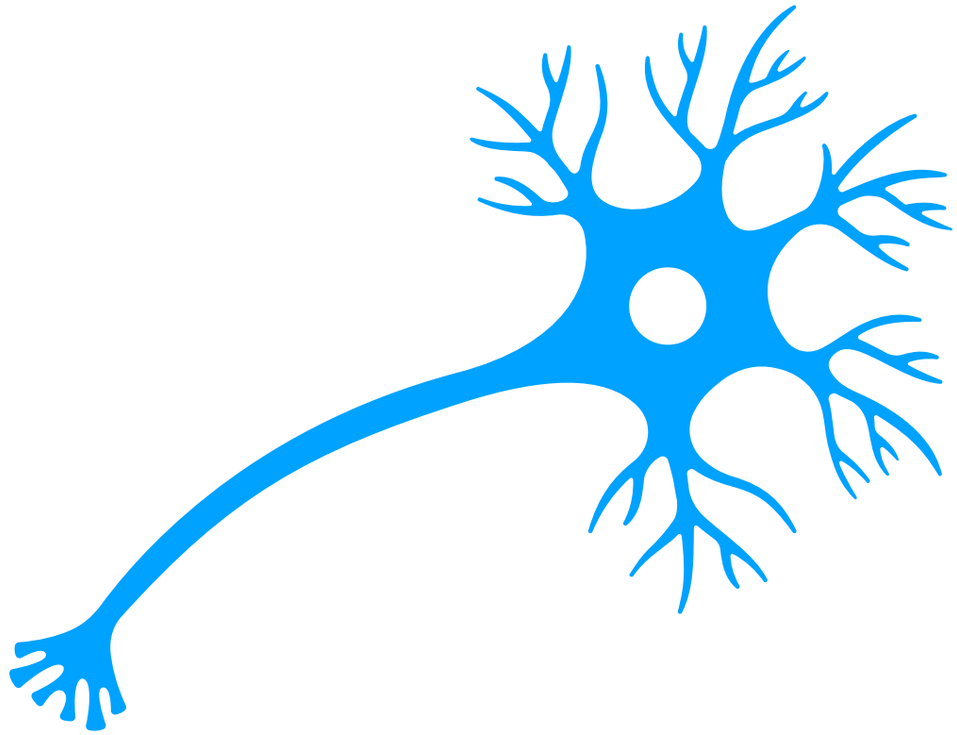
Neither Class

0.114
0.595
= 8593
, 994, 3991]
neither

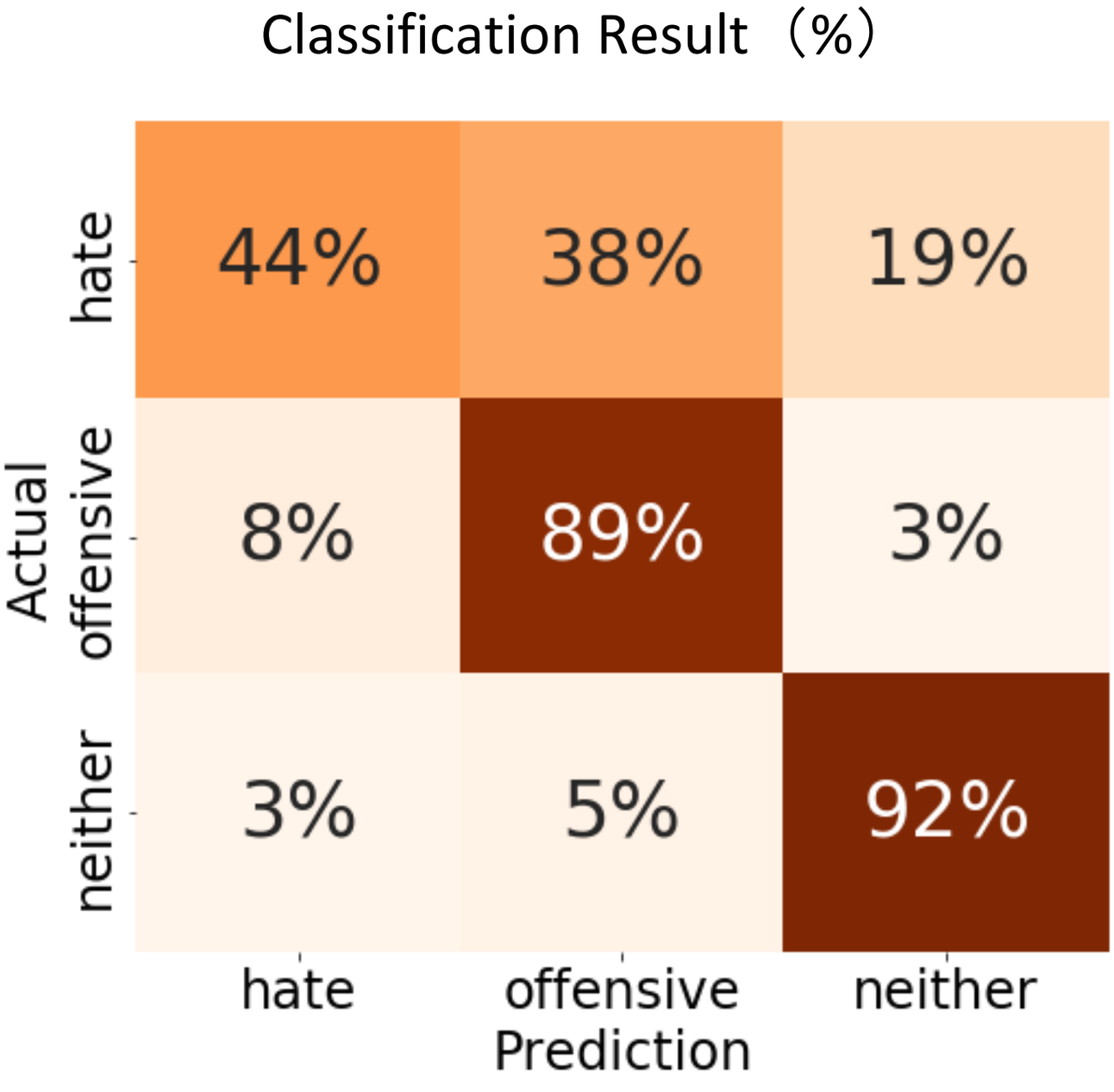
gini = 0.1
samples = 475
value = [4, 21, 450]
class = neither

Hate speech and offensive is distinguished by whether it contains word ‘nigga’

LSTM



Result: Original Data Only



- Similar to decision tree, offensive, and neither are classified well

  f1-score 90%

- Accuracy for hate speech is bad

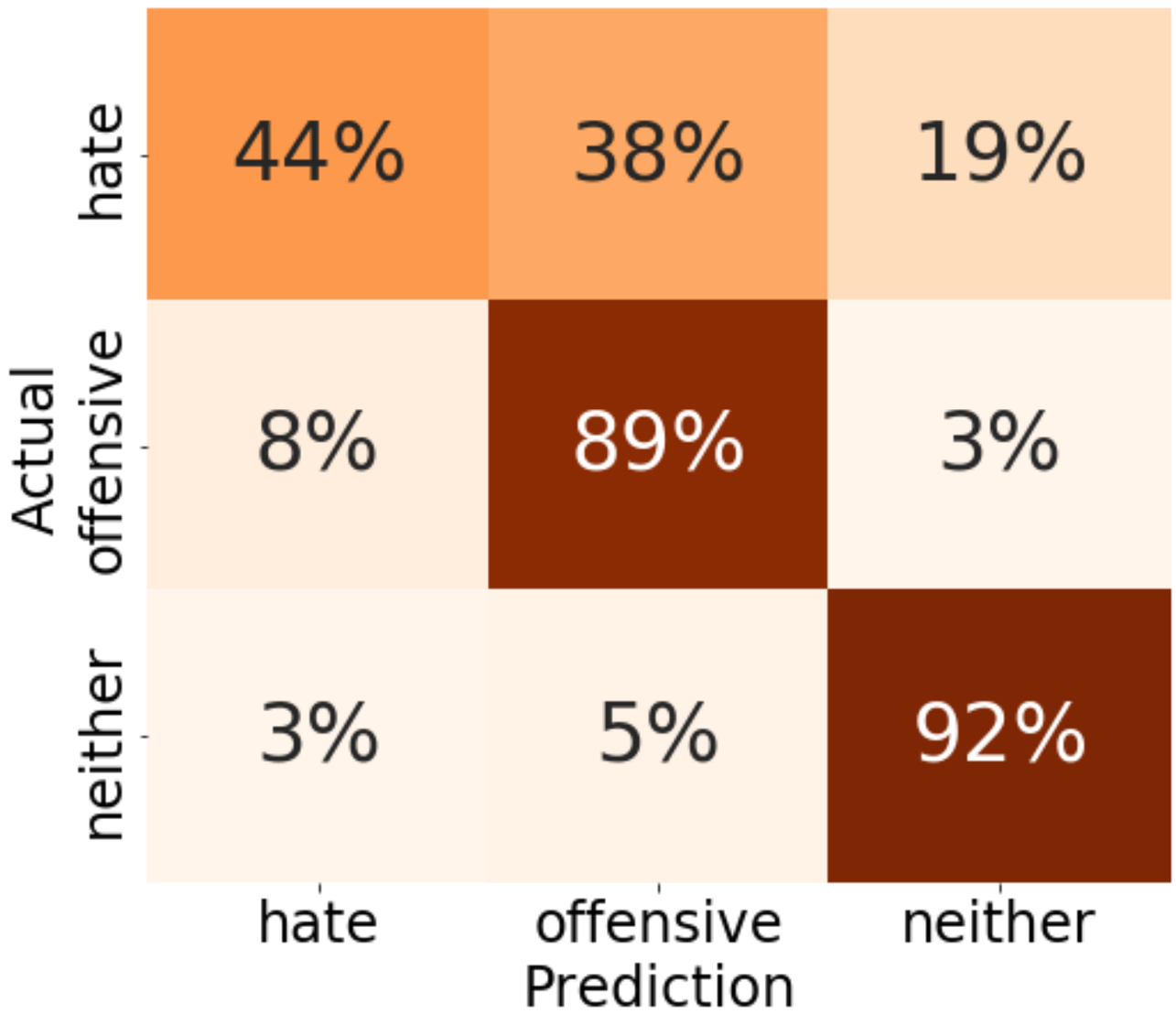
  → 38% is classified as offensive

- f1-macro: 76%

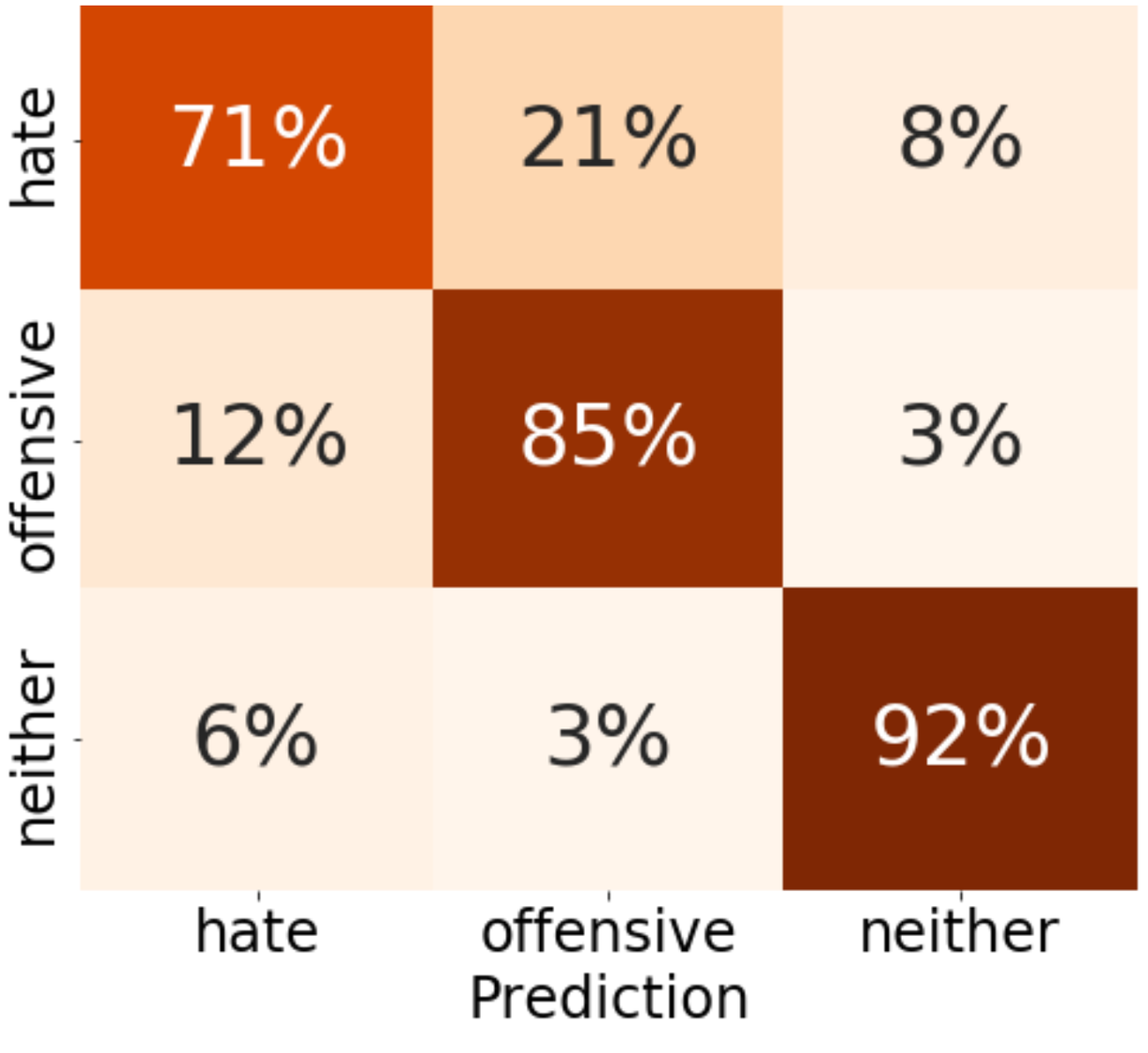
- f1-weighted: 83%

Result: LSTM

Without Data Augmentation

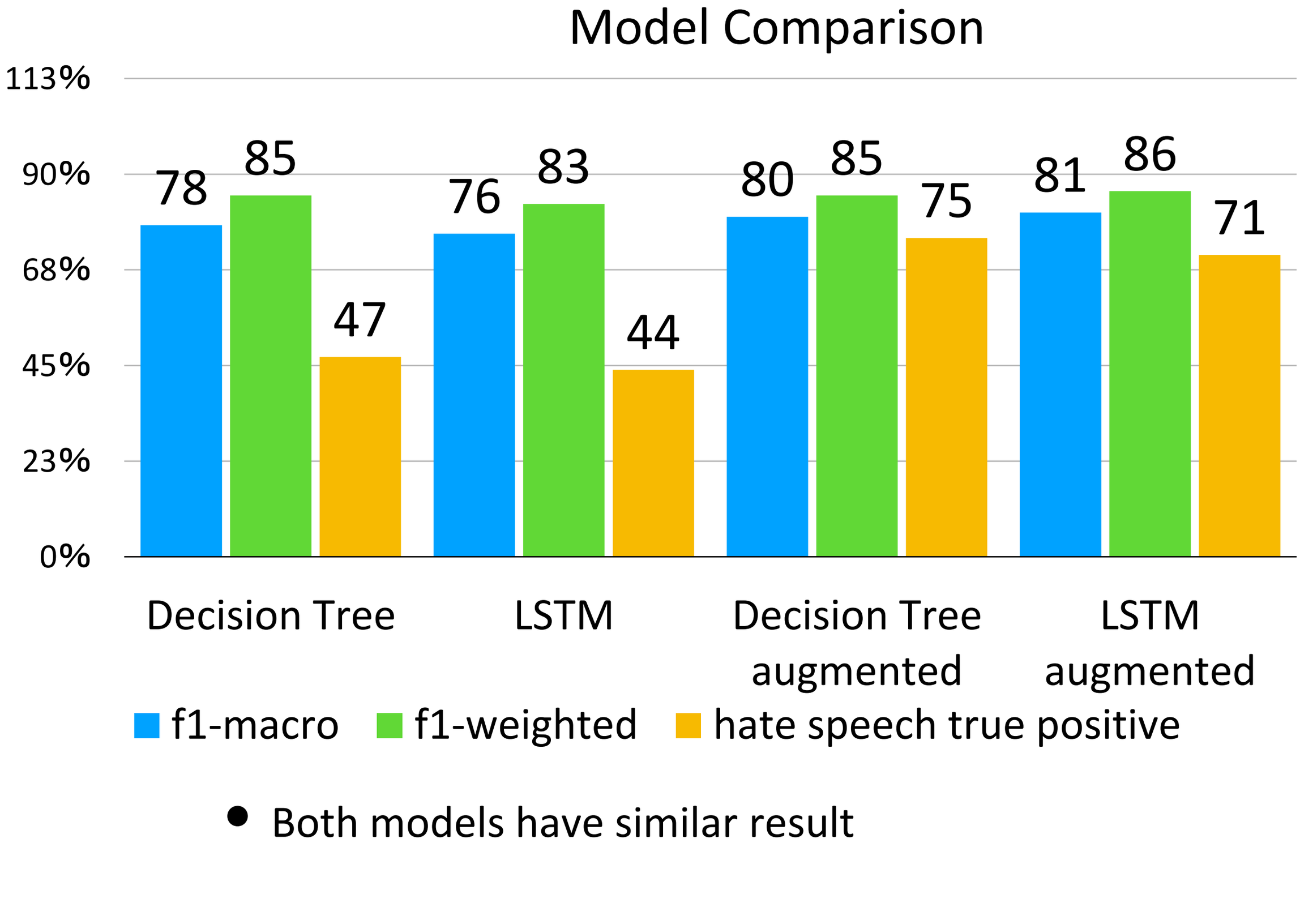


With Data Augmentation



- Again, true positive of hate speech class improved
- False positive of hate speech worthened

- f1-macro: 76% → 81%
- f1-weighted: 83% → 86%



- Both models have similar result

Conclusion

Conclusion

- Trained decision tree and LSTM
- offensive language and neither classes can be detected well

  → both models are good if only trying to detect those two
- There is still problem in classifying hate speech

Conclusion: hate vs offensive

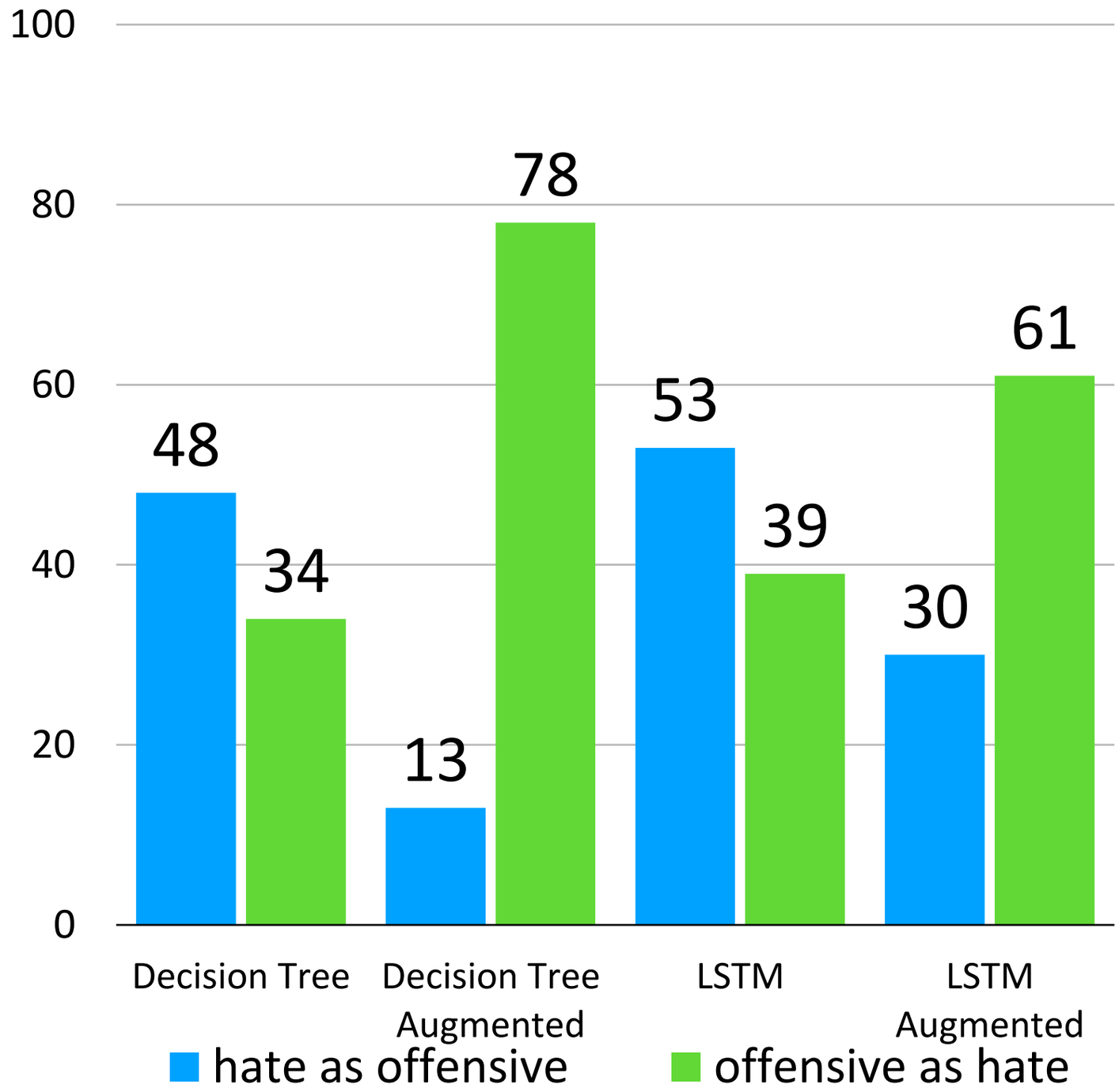
- Models had difficulty distinguishing these two classes

- Original data only

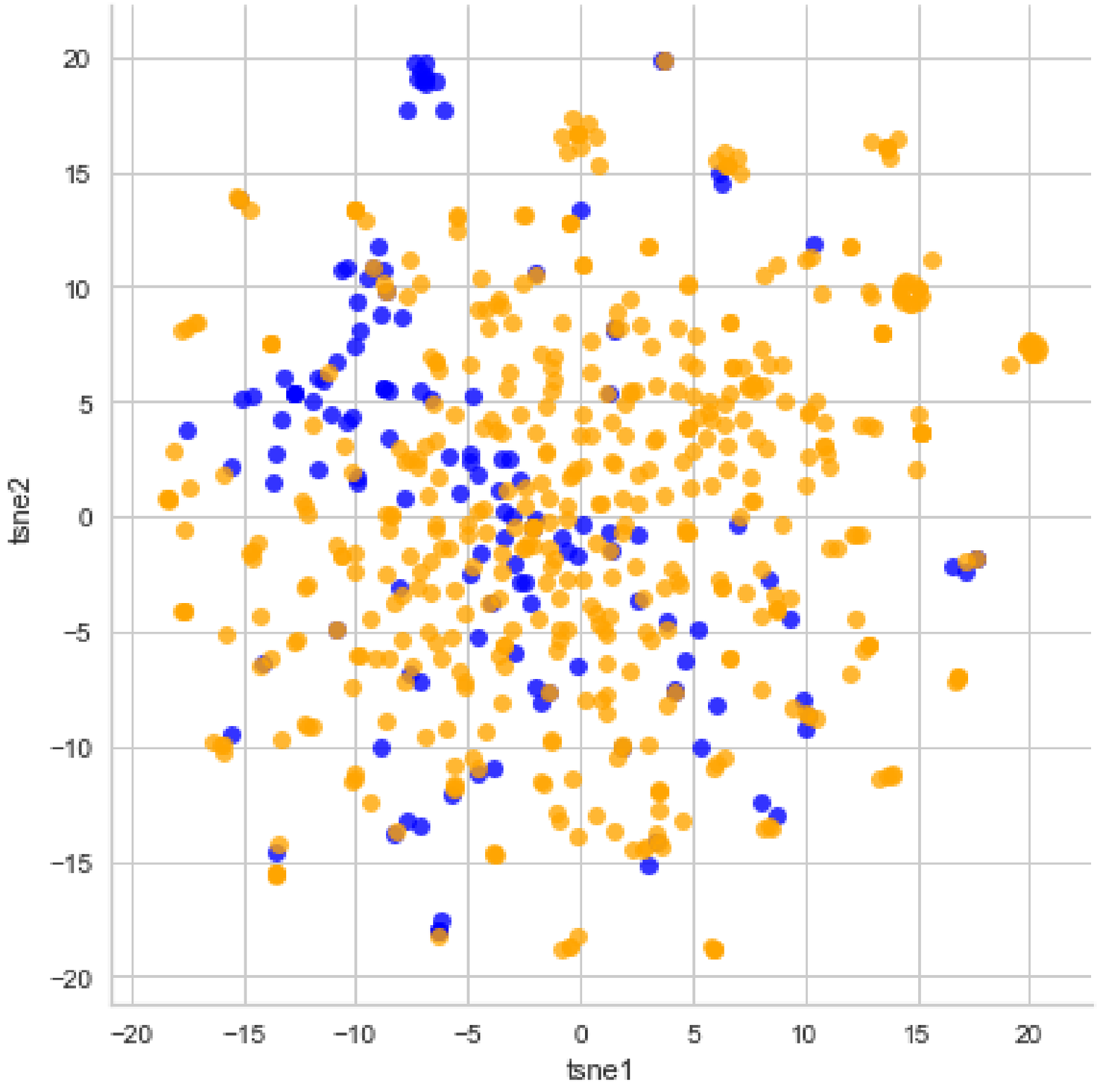
→hate speech likely to be classified as offensive language

- With data augmentation

→offensive language likely to be classified as hate speech



Visualization using TSNE



2 classes are overlapping, maybe this is why the classification did not go well

hate speech

offensive language

Decision Tree

Best Model

Conclusion: Best Model

- It is easier to explain why the model has made its decision

  →Can give explanation when we delete posts from user

  →Less likely to damage user experience

- Performance is satisfactory

  → f1-weighted 85%

Future Work

1. More data
2. Improve faults in decision tree
3. Try more advanced models