FRI Natural Language Processing course 2021
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

Cross-Lingual Offensive Language Identification

Nikolina Grabovica, Selma Halilčević, Matjaž Mav

Abstract
To be add in the final report.

Keywords
Abusive content, offensive language, social media, analysis, detection, training

Advisors: Slavko Žitnik

Introduction

Offensive language or hate speech is commonly understand as communication that expresses hate or encourages violence towards an individual or group based on some characteristic such as race, color, ethnicity, gender, sexual orientation, nationality, religion, or other characteristic [1].

Existing offensive language prevention systems forced online writers to be more creative and these introduce additional challenges. Words and phrases may be obfuscated, for example 'a$$hole', 'ni99a' or 'kill yrslef'. Some of words or phrases might strongly relay on the context where they are used, for example word 'nigga' is associated with negative sentiment, but in some contexts this word may have neutral sentiment. Additionally, it is possible to compose hateful sentence with negations of neutral and positive words.

One of the challenges in research is to define offensive word. We define three broad types of offensive language – profanity, pejorative and obscenity. Profanity is a socially offensive language, which is also defined as cursing and swearing. It shows debasement of someone or something by using impolite, rude and culturally offensive language [2]. A pejorative or slur is the language expressing negative connotation, a low opinion or disrespect to someone or something [3]. An obscenity is a dirty word or phrase expressing possible lewd, bawdy or offensive emotions to someone [4].

In this paper we review and explore the field of the offensive language detection, classification and clustering on different English datasets. The goal of this paper is to analyse a few of datasets and find more granular offensive language clusters among the top level classes that are commonly annotated in datasets.

In the next section we present some of the research studies on offensive language and programming techniques on its detection and prevention. It is followed with Methods section in which we describe datasets we use and processing methodology. Results we obtain from analysis are in Results section, which is followed by Discussion.

The code and results are available on the Github repository matjazmav/fri-2021-nlp-project.

Related Work

In the most of early works simple features, like bag of words, n-grams and character n-grams are used. These simple features are reported to be highly predictive. Additionally Nobata C. et al. [5] show that other simple features like frequency of capitalization, use of non-numerical characters and links can also be used for the offensive language detection.

To detect offensive language in social media to prevent adolescent cyberbullying, Chen Y. et al. [6] propose the Lexical Syntactic Feature (LSF) architecture. This architecture assigns offensive weight value to each sentence by combining its offensiveness of words and the context. Words' offensiveness weight is calculated from its labeling type, which can be profanity, pejorative or obscenity. It is interesting to notice that combination of pejoratives and obscenities is labeled as strongly offensive if more than 80% of their use in a dataset is offensive, while being alone each is classified as a weak offensive word. To label a sentence context as offensive, LSF architecture captures all dependency types between words in the sentence and marks related words as intensifiers, which can describe users or other offensive words.

Another approach to prevent cyberbullying is proposed by Jacobs G. et al. [7] in which they derive positive and negative opinion word ratios and post polarity from subjectivity lexicon features. Subjectivity lexicons provide words' sentiments, which can express words' polarity (positive, negative or neutral), emotions or psychological processes.

Shen J. et al. [8] propose hierarchical clustering method known as Brown clustering. This clustering method has tendency to cluster words of opposite sentiment together, for example words like ’good’ and ’bad’ are clustered in the same cluster. In order to better generalize word representations Tian Z. et al. [9] apply Brown clustering method separately on the positive and negative sentiment data and later combine the information into a single complex feature. They show that the new information is beneficial to both simple and deep classifiers.

More recent research focus more on the deep learning methods. These deep representations of text (word, paragraph or document) are refereed to as embeddings. As we pointed out in the introduction, the context information of where the phrase is used is usually very important. The simplest approach to introduce the context information to the embedding is to average word embeddings of the entire phrase or sentence [5].

Martinc M. and Pollak S. [10] combine n-grams and convolutional neural network (CNN) for author profiling language variety classification. Inputs for the CNN are word bound character n-grams of sizes between three and five. They train six different classification models, where each model corresponds to one language group. Accuracy and $F1$-score for all language groups using TF-IDF are 0.9981 and 0.9981, respectively. They state that proposed system performed well for all binary predictions.

Methodology

Terminology

RNN

Recurrent Neural Networks (RNNs) are a form of machine learning algorithm that is ideal for sequential data such as text, speech, audio, video, etc. The power of neural networks comes from their ability to find data representations that are valuable for classification. RNNs are a specific type of neural network, which can be thought of as a completely connected neural network that contains a refactoring of some of its layers into a loop. That loop is typically an iteration over the addition or concatenation of two inputs, matrix multiplication, and a non-linear function. RNNs can be trained using variants of the backpropagation (the algorithm used to find optimal weights in a neural network by performing gradient descent) such as BPTT (backpropagation through time), to update the network weights in every layer.

BERT

In the domain of computer vision, researchers have repeatedly shown the benefit of transfer learning (pretraining a neural network model on a known task), and recent studies shown that a resembling technique can be useful in natural language tasks. BERT (Bidirectional Encoder Representations from Transformers) is an imminent paper published by researchers at Google AI Language. BERT’s innovation is based on applying the bidirectional training of Transformer a popular attention model that learns contextual relations between words or sub-words in a document, to language modeling.

Data

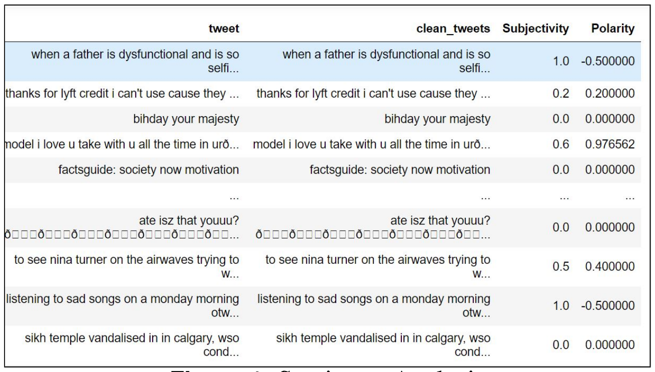
To perform exploratory analysis of the use of offensive language in social media, we test our methods on several online available datasets, listed below:

- Dataset collected by [11] from Twitter with hate, offensive and neither tags in English. Dataset contains file with labeled data and refined n-gram speech lexicon.
- Dataset of Twitter API IDs and tags - sexist, racist and not. It is provided in English by [12].
- Dataset with tags of benovelent and hostile sexism from Twitter in English [13].
- Multilingual and expert based dataset – CONAN obtained from Facebook posts. Tags are binary (islamophobic or not), multi-topic (culture, economics, crimes, rapism, terrorism, women oppression, history, other/-generic) [14].
- Multilingual dataset collected from Twitter with tags on hostility, directness, target attribute and group [15].
- MMHS150K dataset containing multiple categories, such as racist, sexist, homophobic, religion based attack, attack to other community in English from Twitter [16].

Methods

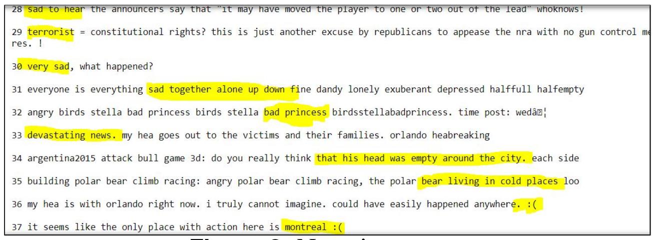
Sentiment Analysis

In ”sentiment-analysis-text-classification-baseline-model.py” we can find one baseline model for text classification with TensorFlow Keras. First the Sentiment Analysis was done. I have uploaded the non-processed data set (dataset6) with 31962 tweets. To clean the tweets from URLs, hyperlinks, mentions, stop-words, etc., we need to apply regular expressions on the non-processed text. After we cleaned the data we start with Sentiment analysis. Sentiment analysis is basically the process of determining the attitude or the emotion of the writer, i.e., whether it is positive, negative or neutral. To do that we used sentiment function of TextBlob which returns two properties polarity and subjectivity. Polarity is float which lies in the range of [-1,1] where 1 means positive statement and -1 means a negative statement. Subjective is also a float which lies in the range of [0,1]. Subjective sentences generally refer to personal opinion, emotion or judgment whereas objective refers to factual information. After we cleaned the tweets and got the Polarity and Subjectivity, let’s check data from our data set. At the Figure 1, we can see values for specific tweet from our data set, for the tweets which has the polarity grater than 0 tweets are consider positive, for ones less than 0 they are consider negative, and ones equal to zero are labeled as neutral. In order to review only the negative tweets we printed out the tweets using the negative polarity which we calculated before. At the Figure 2, we can spot the tweets that are considered to have negative significance.



	tweet	clean_tweets	Subjectivity	Polarity
	when a father is dysfunctional and is so selfi...	when a father is dysfunctional and is so selfi...	1.0	-0.500000
	thanks for lyft credit i can't use cause they ...	thanks for lyft credit i can't use cause they ...	0.2	0.200000
	bihday your majesty	bihday your majesty	0.0	0.000000
	nodel i love u take with u all the time in urð...	model i love u take with u all the time in urð...	0.6	0.976562
	factsguide: society now motivation	factsguide: society now motivation	0.0	0.000000
	...	...	...	...
	ate isz that youuu? ð□□□ð□□□□ð□□□□ð□□□□ð□□□□ð□□□□ð□□...	ate isz that youuu? ð□□□ð□□□□ð□□□□ð□□□□ð□□□□ð□□□□ð□□...	0.0	0.000000
	to see nina turner on the airwaves trying to w...	to see nina turner on the airwaves trying to w...	0.5	0.400000
	listening to sad songs on a monday morning otw...	listening to sad songs on a monday morning otw...	1.0	-0.500000
	sikh temple vandalised in in calgary, wso cond...	sikh temple vandalised in in calgary, wso cond...	0.0	0.000000

Figure 1. Sentiment Analysis



```
28 sad to hear the announcers say that "it may have moved the player to one or two out of the lead" whoknows!

29 terrorist = constitutional rights? this is just another excuse by republicans to appease the nra with no gun control me
res. !

30 very sad, what happened?

31 everyone is everything sad together alone up down fine dandy lonely exuberant depressed halffull halfempty

32 angry birds stella bad princess birds stella bad princess birdsstellabadprincess. time post: wedâ¦

33 devastating news. my hea goes out to the victims and their families. orlando heabreaking

34 argentina2015 attack bull game 3d: do you really think that his head was empty around the city. each side

35 building polar bear climb racing: angry polar bear climb racing, the polar bear living in cold places loo

36 my hea is with orlando right now. i truly cannot imagine. could have easily happened anywhere. :(

37 it seems like the only place with action here is montreal :(
```

Figure 2. Negative tweets

Text Classification

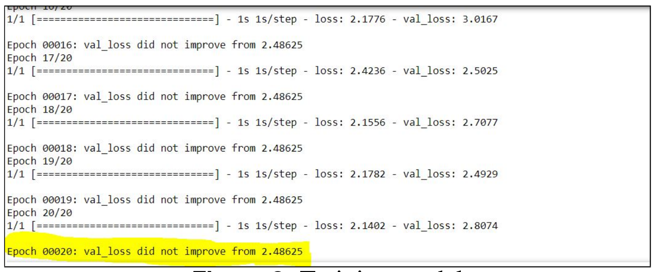
In order to continue with text classification, first we counted how many times each word is repeated in negative tweets. Words like empty, attack, sad emojis, swear words, etc., are in the largest number of repetition. To create and train a model we used TensorFlow. Each sequence needs to have same length, so we are mapping them to the same sequence size. We are going to split a dataset into a train / test split, we are going to use 20 percent test set. Train sentences are first elements in text feature and test sentences are the remaining 20 percent. Afterwards we are going to use Tokenizer class from Keras to Tokenize the train sentences, and we are going to create sequences from our Tokenizer based on the indexes from word index, and check sentences produces by training data. For our model we are going to use embedded layers and LSTM architecture, embeding layers are mapping each word to a vector of a fix sized size with a real value elements.

Training Classic Classifiers

We use simple classic classifiers to train our data on offensive language prediction. These classifiers are Logistic regression, Support Vector Machine (SVM) and Multinomial Naive Bayes. We train each classifier on the unique combined dataset from datasets [11], [14], which are first preprocessed to remove tags, urls, exclamation marks and similar. Dataset contains six classes of offensive language, which are None, Rasism/sexism, Hate speech, Offensive language, Profanity and Islamophobia. To obtain models, we calculate *ngram* and $tf-idf$ values from training data and process them with the classifier. We evaluate our models in terms of accuracy, precision, f_1-score, confusion matrix and ROC curve. We present performance of our classifiers in the Results section.

Neural Machine Translation

Sequence-to-Sequence (seq2seq) models are used for a multiplicity of NLP tasks, we are going to use it in purpose to translate given sentences from one language to another. Seq2seq model have an encoder and decoder, these parts are essentially two different recurrent neural networks (RNN) models combined into one big network. After we did a text pre-processing we start to build a model. For train and test datasets we will encode Slovenian sentences as the input sequences and English sentences as the target sequences. In order to have Seq2Seq architecture for the encoder, we will use an embedding layer and an LSTM layer, and for the decoder, we will use another LSTM layer followed by a dense layer, with the RMSprop optimizer, for the loss function we used 'sparse-categorical-crossentropy' because the function allows us to use the target sequence as is. We will train it for 20 epochs, batch size of 512 with a validation split of 20 precent. At the Figure 3, we can observe that validation loss did not improve from 2.48625 which was at 15/20 epoch.



```
1/1 [==============================] - 1s 1s/step - loss: 2.1776 - val_loss: 3.0167

Epoch 00016: val_loss did not improve from 2.48625
Epoch 17/20
1/1 [==============================] - 1s 1s/step - loss: 2.4236 - val_loss: 2.5025

Epoch 00017: val_loss did not improve from 2.48625
Epoch 18/20
1/1 [==============================] - 1s 1s/step - loss: 2.1556 - val_loss: 2.7077

Epoch 00018: val_loss did not improve from 2.48625
Epoch 19/20
1/1 [==============================] - 1s 1s/step - loss: 2.1782 - val_loss: 2.4929

Epoch 00019: val_loss did not improve from 2.48625
Epoch 20/20
1/1 [==============================] - 1s 1s/step - loss: 2.1402 - val_loss: 2.8074

Epoch 00020: val_loss did not improve from 2.48625
```

Figure 3. Training model

Results

Fig. 6a shows the distribution of classes present in the combined dataset. Training data contains 70%, and test data 30%, randomly generated, samples from the dataset.

We present confusion matrices as the performance measurement of the three classification model on Fig. 4 and 5.

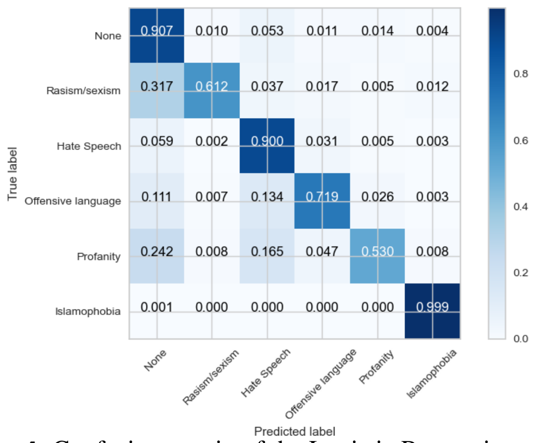
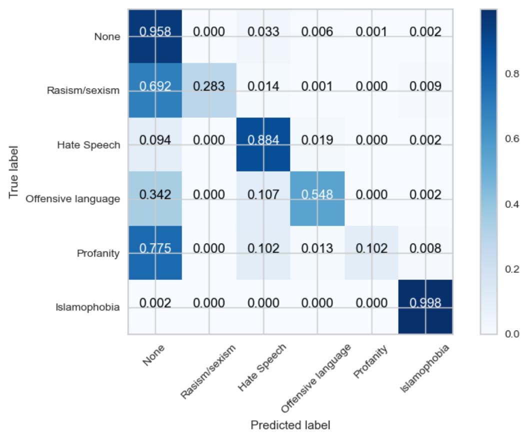
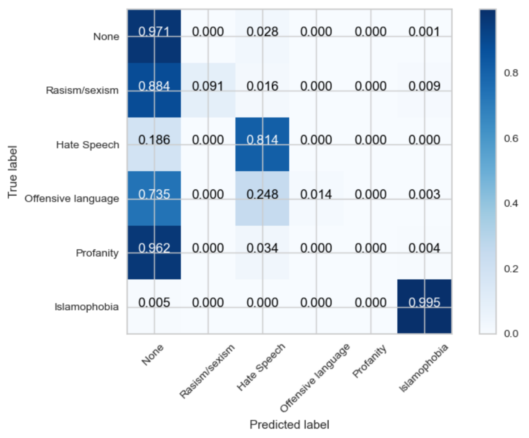


Figure 4. Confusion matrix of the Logistic Regression model on the combined dataset.



(a) Confusion matrix of the SVM model on the combined dataset.



(b) Confusion matrix of the Multinomial Naive Bayes model on the combined dataset.

Figure 5. SVM and Multinomial Naive Bayes confusion matrices.

Another one performance measure ROC curve is shown on Fig. 6b, 6c and 6d for Logistic regression, SVM and Multinomial Naive Bayes models, respectively. We, also, show the performance of the classifiers in terms of accuracy, precision and f_1-score in Tab. 1.

Table 1. Accuracy, precision and f_1-score for all three classification models.

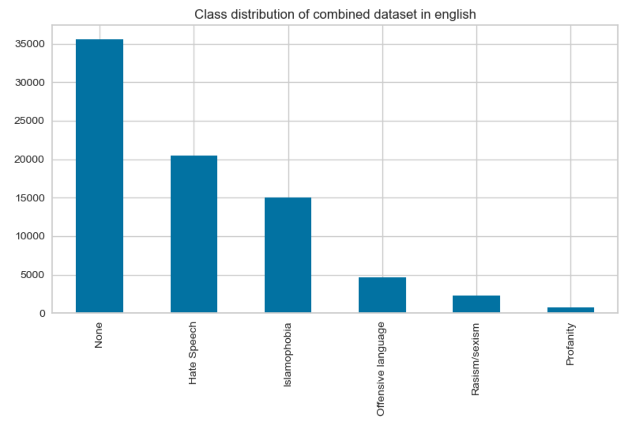
Classifier	*Accuracy*	*Precision*	*F_1-score*
Logistic Regression	0.9000	0.77275	0.7705
SVM	0.8951	0.8727	0.6776
Multinomial NB	0.8445	0.7683	0.4832

Discussion

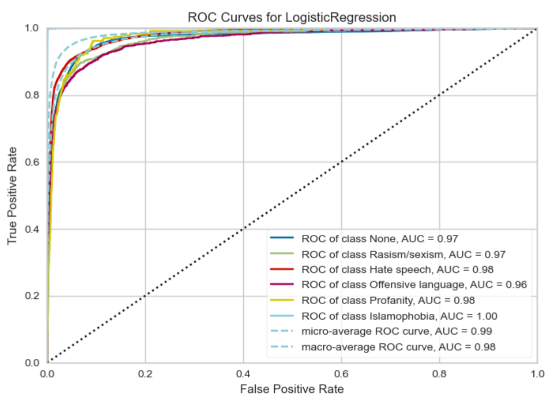
We observe from Fig. 6 that Logistic regression model is the best separators between true positive and false positive samples. It is followed by the SVM model. Using these two models even *profanity* class was correctly classified with around 0.98 AUC. Multinomial Naive Bayes dractically failed for good offensve language classification, which proves confusion matrix, Fig. 5b, ROC curve, Fig. 6d and metrics in Tab. 1. In each of these measures, Multinomial Naive Bayes is outperformed by other two models.

Conclusion

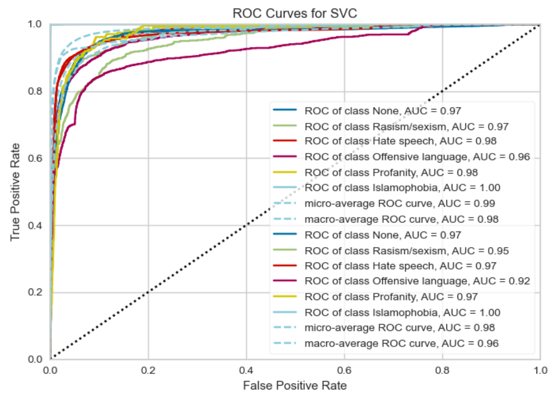
TODO



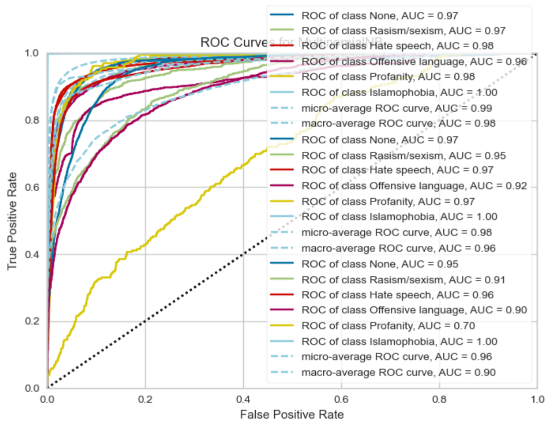
(a) Class distribution present in the dataset.



(b) ROC curve on Logistic Regression model.



(c) ROC curve on SVM model.



(d) ROC curve on Multinomial Naive Bayes model.

Figure 6. Class distribution and ROC analysis of the three classic classifiers on the large combined dataset.

References

[1] Anna Schmidt and Michael Wiegand. A survey on hate speech detection using natural language processing. In *Proceedings of the fifth international workshop on natural language processing for social media*, pages 1–10, 2017.

[2] *The Monthly Supplement: a current biographical reference service*. Number v. 1-2. A. N. Marquis., 1940.

[3] Collins English Dictionary. Pejorative - definition, 2012.

[4] Vocabulary.com. Obscenity - definition.

[5] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *Proceedings of the 25th international conference on world wide web*, pages 145–153, 2016.

[6] Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. Detecting offensive language in social media to protect adolescent online safety. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80. IEEE, 2012.

[7] Gilles Jacobs, Cynthia Van Hee, and Véronique Hoste. Automatic classification of participant roles in cyberbullying: Can we detect victims, bullies, and bystanders in social media text? *Natural Language Engineering*, pages 1–26, 2020.

[8] Judong Shen, Shing I Chang, E Stanley Lee, Youping Deng, and Susan J Brown. Determination of cluster number in clustering microarray data. *Applied Mathematics and Computation*, 169(2):1172–1185, 2005.

[9] Zuoyu Tian and Sandra Kübler. Offensive language detection using brown clustering. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5079–5087, 2020.

[10] Matej Martinc and Senja Pollak. Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, 25(5):607–632, 2019.

[11] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the*

11th International AAAI Conference on Web and Social Media, ICWSM '17, pages 512–515, 2017.

[12] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California, June 2016. Association for Computational Linguistics.

[13] Akshita Jha and Radhika Mamidi. When does a compliment become sexist? analysis and classification of ambivalent sexism using twitter data. In *Proceedings of the Second Workshop on NLP and Computational Social Science*, pages 7–16, 2017.

[14] Yi-Ling Chung, Elizaveta Kuzmenko, Serra Sinem Tekiroglu, and Marco Guerini. Conan–counter narratives through nichesourcing: a multilingual dataset of responses to fight online hate speech. *arXiv preprint arXiv:1910.03270*, 2019.

[15] Nedjma Ousidhoum, Zizheng Lin, Hongming Zhang, Yangqiu Song, and Dit-Yan Yeung. Multilingual and multi-aspect hate speech analysis. In *Proceedings of EMNLP*. Association for Computational Linguistics, 2019.

[16] Raul Gomez, Jaume Gibert, Lluis Gomez, and Dimosthenis Karatzas. Exploring hate speech detection in multimodal publications. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1470–1478, 2020.