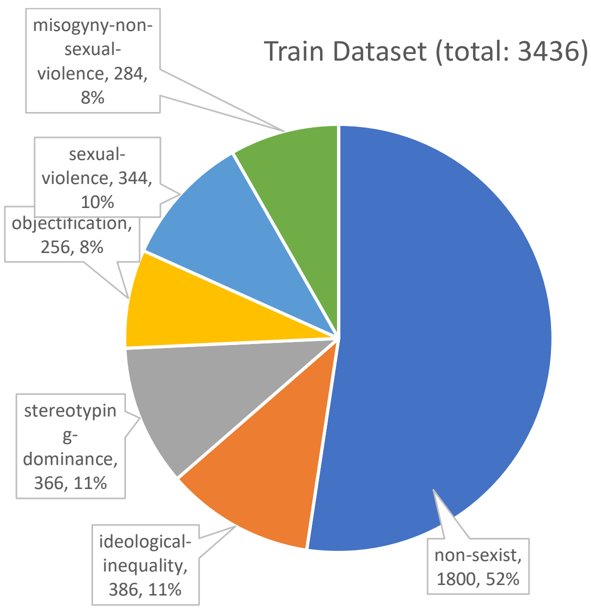
Sexism Auto-detection in Social Media Using Deep Learning Methods

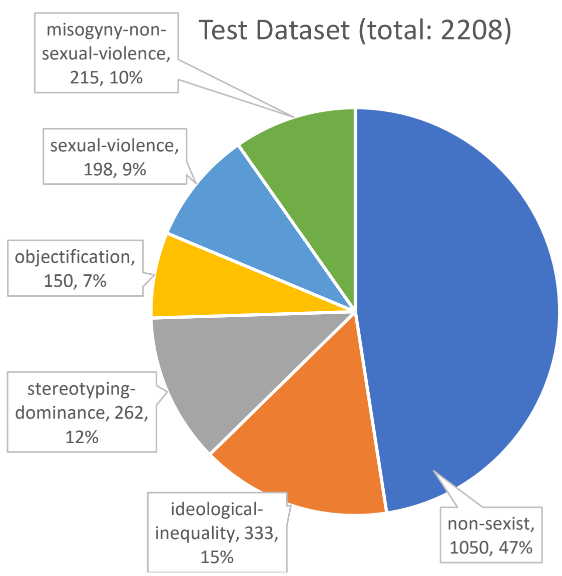
Dandi Yu

Introduction and Objectives of the project

- The Objective of the project is to implement English sexism content (tweets and gabs) classification with only very limited dataset (train on 3000 texts) by using deep learning methods, and to explore ways to improve the prediction performance.
- The unique point of the project: create a model to perform good sexism prediction with very limited training dataset
- Dataset: EXIST dataset shared for IberLEF2021 (http://nlp.uned.es/exist2021/).
- Task 1: binary classification with labels ‘sexist’ and ‘non-sexist’
- Task 2: multi-class classification with labels as ‘non-sexist’, ‘ideological-inequality’, ‘stereotyping-dominance’, ‘objectification’, ‘sexual-violence’, ‘misogyny-non-sexual-violence’
- Basic Models: using various deep learning methods Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), BERT / DistilBERT transformers.
- Advanced Models: improve the performance by using all hidden layers of the BERT encoders, and external knowledge-based features in addition to the “Basic Models”.

EXIST Dataset (Task 2 labels)





Class	Examples (raw text before pre-processing)
non-sexist	"@user woaw- you're the most beautiful person i've ever seen"
ideological-inequality	"@user Feminism is cancer! Happy new year"
stereotyping-dominance	"call me sexist but i've never seen a girl eat a whole large dominos pizza by herself"
objectification	"@user Wow, your skirt is very short. What is it's length? 5 inch or more?"
sexual-violence	"@user you look like a bitch"
misogyny-non-sexual-violence	"@user This why I hate women"

Summary of the models

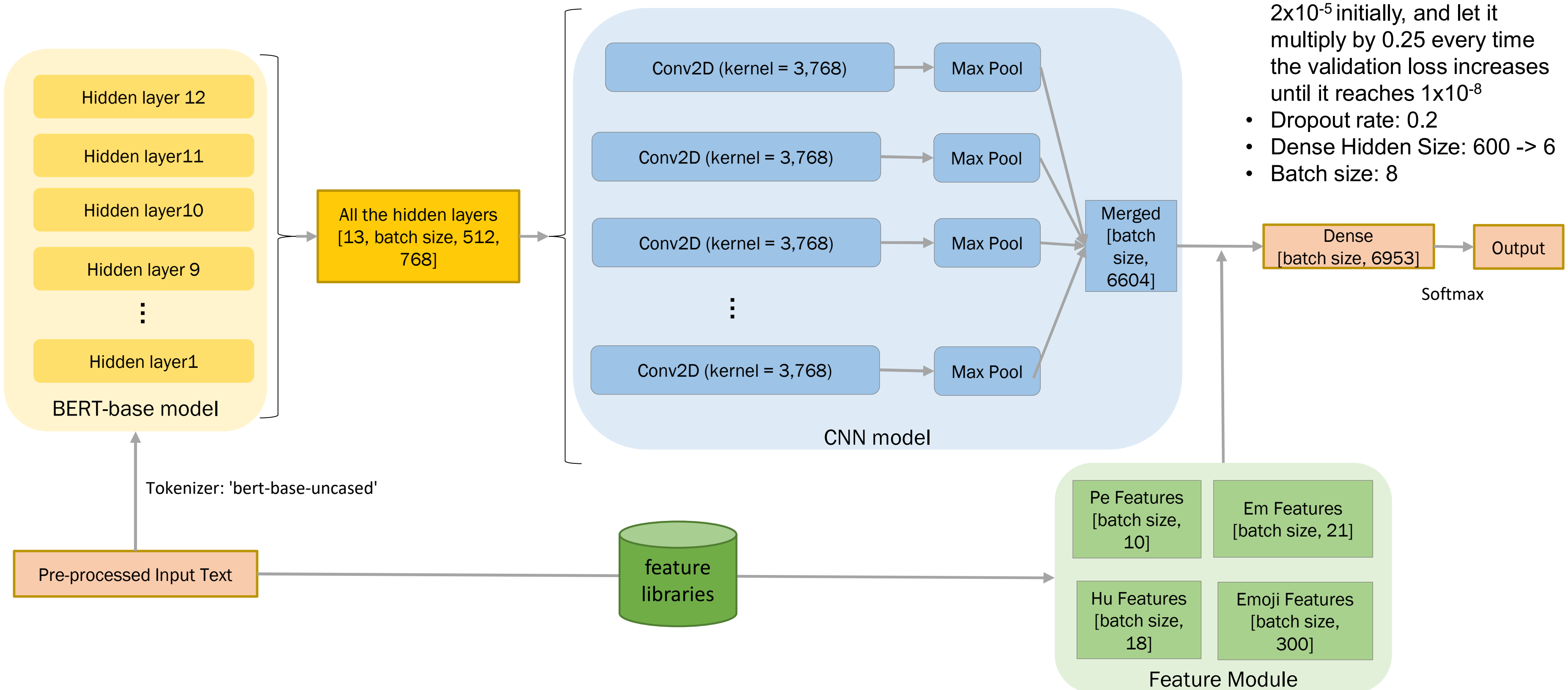
Model	
/	NBOW
/	NBOW with GLOVE (baseline)
DistilBERT +	Sequence Classification
	NBOW
	LSTM
	Single-filter CNN
	Multi-filter CNN
Bert +	Sequence Classification
	NBOW
	LSTM
	Single-filter CNN
	Multi-filter CNN
	CNN 12-layers + BiLSTM-Attention Features
	CNN 12-layer + Features
	CNN last 4-layer + Features

Basic Models

Advanced Models

“BERT+CNN 12-layer+Features” model architecture for Task 2

- Optimizer: Adam
- Loss function: categorical cross entropy
- Early Stop: learning rate $2x10^{-5}$ initially, and let it multiply by 0.25 every time the validation loss increases until it reaches $1x10^{-8}$
- Dropout rate: 0.2
- Dense Hidden Size: 600 -> 6
- Batch size: 8



External Knowledge-based Features

- Perspective API: Google's Perspective API is an API uses machine learning models to analyze and score the targeted textual contents in nine dimensions of emotional concepts for English texts: (toxicity, severe toxicity, identity attack, insult, profanity, threat, sexual explicit, obscene, and flirtation.) The score in each dimension is scored in the range between 0 and 1. For each sentence of the tweet, a **9-dimensional** vector is created.
- HurtLex (Hu): HurtLex is a lexicon of offensive, aggressive, and hateful words in over 50 languages. It has a 2-level structure of 'conservative' or 'inclusive' and it has divided into 17 categories. I have utilized 9 categories out of them as these are more related to our problem. For each sentence of the tweet, a **18-dimension vector** is created as we consider both 2 levels of 'conservative' or 'inclusive'.
- Empath (Em): Empath is a tool to analyze text across lexical categories. Empath draws connotations between words and phrases by deep learning a neural embedding across more than 1.8 billion words of modern fiction. 21 categories were used in my model (sexism, violence, money, valuable, domestic work, hate, aggression, anticipation, crime, weakness, horror, swearing terms, kill, sexual, cooking, exasperation, body, ridicule, disgust, anger, and rage). A **21-dimension vector** is created for each sentence of the tweet.
- Emoji: a pre-trained **300-dimension** emoji2vec semantic embedding has been used to transfer the emojis to this vector as another feature.

Training result in each epoch

task2

```
model = train_model(train_dataloader, valid_dataloader, 20)
```

```
Some weights of the model checkpoint at bert-base-uncased were not used when initializing BertModel: ['cls.predictions.decoder.weight', 'cls.predictions.transform.dense.bia
- This IS expected if you are initializing BertModel from the checkpoint of a model trained on another task or with another architecture (e.g. initializing a BertForSequenc
- This IS NOT expected if you are initializing BertModel from the checkpoint of a model that you expect to be exactly identical (initializing a BertForSequenceClassificatio
/usr/local/lib/python3.7/dist-packages/transformers/optimization.py:310: FutureWarning: This implementation of AdamW is deprecated and will be removed in a future version.
  FutureWarning,
Epochs: 1 | LR: [2e-05] | Train Loss:  1.308          | Train Accuracy:  0.537          | Val Loss:  1.221          | Val Accuracy:  0.546
Epochs: 2 | LR: [2e-05] | Train Loss:  0.940          | Train Accuracy:  0.656          | Val Loss:  1.048          | Val Accuracy:  0.604
Epochs: 3 | LR: [2e-05] | Train Loss:  0.634          | Train Accuracy:  0.780          | Val Loss:  1.090          | Val Accuracy:  0.616
Epochs: 4 | LR: [5e-06] | Train Loss:  0.301          | Train Accuracy:  0.903          | Val Loss:  1.209          | Val Accuracy:  0.641
Epochs: 5 | LR: [1.25e-06] | Train Loss:  0.204          | Train Accuracy:  0.939          | Val Loss:  1.190          | Val Accuracy:  0.646
Epochs: 6 | LR: [1.25e-06] | Train Loss:  0.162          | Train Accuracy:  0.954          | Val Loss:  1.226          | Val Accuracy:  0.631
Epochs: 7 | LR: [3.125e-07] | Train Loss:  0.146          | Train Accuracy:  0.955          | Val Loss:  1.214          | Val Accuracy:  0.640
Epochs: 8 | LR: [3.125e-07] | Train Loss:  0.144          | Train Accuracy:  0.961          | Val Loss:  1.234          | Val Accuracy:  0.653
Epochs: 9 | LR: [7.8125e-08] | Train Loss:  0.131          | Train Accuracy:  0.965          | Val Loss:  1.252          | Val Accuracy:  0.644
Epochs: 10 | LR: [1.953125e-08] | Train Loss:  0.129          | Train Accuracy:  0.966          | Val Loss:  1.223          | Val Accuracy:  0.644
Epochs: 11 | LR: [1.953125e-08] | Train Loss:  0.131          | Train Accuracy:  0.964          | Val Loss:  1.214          | Val Accuracy:  0.637
Epochs: 12 | LR: [1.953125e-08] | Train Loss:  0.127          | Train Accuracy:  0.965          | Val Loss:  1.236          | Val Accuracy:  0.649
```

Experiment Results

Best Model for Task 1

Table II. TASK 1 RESULT

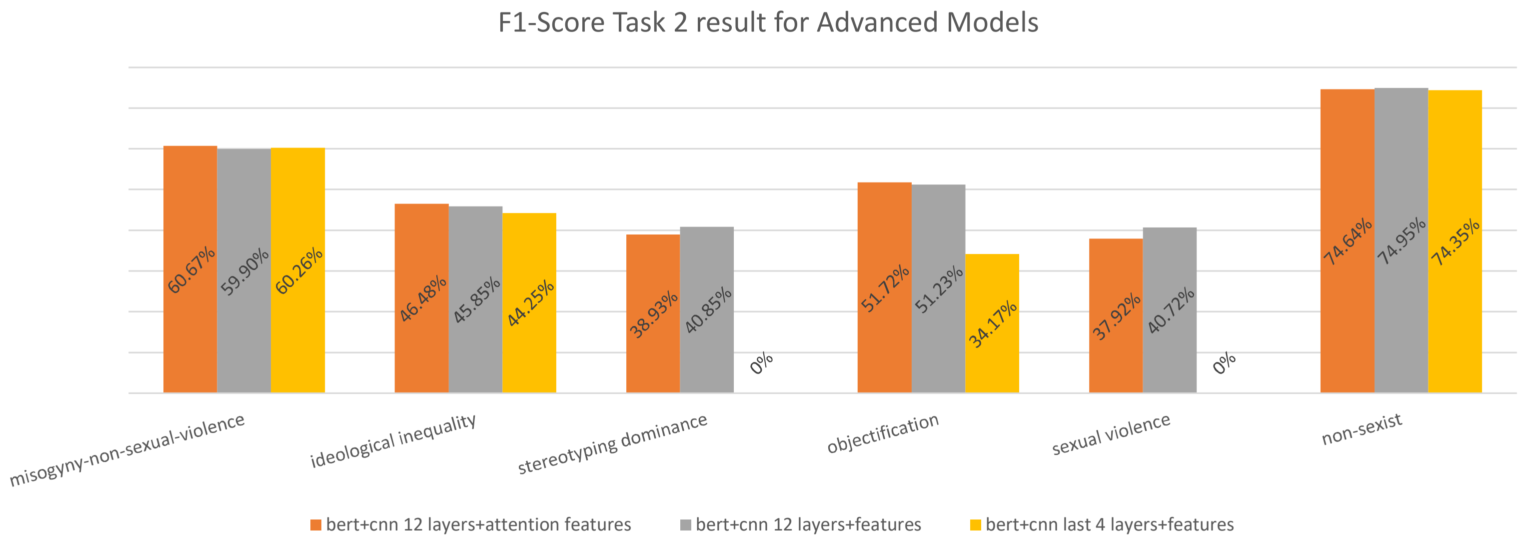
Model		Accuracy	F1 Score	Precision	Recall
/	NBOW	0.4923	0.3724	0.6005	0.5149
/	NBOW with GLOVE (baseline)	0.4724	0.4676	0.4769	0.4781
DistilBERT +	Sequence Classification	0.6997	0.6898	0.7124	0.6926
	NBOW	0.7319	0.7316	0.7316	0.7321
	LSTM	0.7065	0.7064	0.7104	0.7092
	Single-filter CNN	0.7355	0.7334	0.7334	0.733
	Multi-filter CNN	0.745	0.7415	0.7491	0.7411
Bert +	Sequence Classification	0.3365	0.2799	0.2635	0.3511
	NBOW	0.726	0.7259	0.7265	0.727
	LSTM	0.7278	0.7275	0.728	0.7275
	Single-filter CNN	0.7197	0.7186	0.719	0.7184
	Multi-filter CNN	0.7378	0.7378	0.7388	0.7391
	CNN 12-layers + BiLSTM-Attention Features	**0.7708**	**0.77**	**0.7704**	**0.7698**
	CNN 12-layer + Features	0.7577	0.7572	0.7571	0.7572
	CNN last 4-layer + Features	0.7541	0.7532	0.7536	0.753
Competition Result (Rodríguez-Sánchez et al., 2021) [28]					
task1 SINAI TL 3.tsv en (Best Model)		0.7772	0.7747	0.7805	0.7739

Best Model for Task 2

Table III. TASK 2 RESULT

Model		Accuracy	F1 Score	Precision	Recall
/	NBOW	0.4755	0.1074	0.0793	0.1667
/	NBOW with GLOVE (baseline)	0.4755	0.1074	0.0793	0.1667
DistilBERT +	Sequence Classification	0.5593	0.4458	0.469	0.4409
	NBOW	0.5630	0.4820	0.4955	0.4783
	LSTM	0.5974	0.4827	0.5173	0.462
	Single-filter CNN	0.5865	0.4774	0.5151	0.4721
	Multi-filter CNN	0.5915	0.5152	0.5154	0.5333
Bert +	Sequence Classification	0.1476	0.047	0.0353	0.164
	NBOW	0.5806	0.4921	0.4934	0.507
	LSTM	0.5919	0.5083	0.4984	0.5276
	Single-filter CNN	0.5947	0.5177	0.5093	0.5322
	Multi-filter CNN	0.5806	0.4665	0.5077	0.4561
	CNN 12-layer + BiLSTM-Attention Features	0.6078	0.5173	0.5166	0.5218
	CNN 12-layer + Features	**0.6173**	**0.5225**	**0.5329**	**0.5156**
	CNN last 4-layer + Features	0.5675	0.355	0.3347	0.3935
Competition Result (Rodríguez-Sánchez et al., 2021) [28]					
task2 LHZ 1.tsv en (Best Model)		0.6336	0.5604	0.5512	0.5742

Experiment Results



Experiment Results

- "non-sexist" tweets detection has the best recall value, about 77% of them are correct.
- The classes "ideological-inequality" and "sexual-violence" have the second and third best prediction, about 55.56% and 52.02% of them are predicted correctly.
- The classes "stereotyping-dominance", "misogyny-non-sexual-violence" and "objectification" have low recall in descending order, 49.62%, 40% and 36% respectively.
- By manual inspecting of the misclassifying items, it seems more subtle classes such as "objectification" tends to be more likely to be predicted wrongly as "non-sexist" comparing with more explicit class such as "sexual-violence". I think "objectification" type of content requires more context to be classified correctly, while other text from more explicit class such as "sexual-violence" involves swear words can be more easily detected.

Conclusion and Future work

- Here, we proposed a transfer learning approach by using pre-trained BERT model combining with CNN and external knowledge-based features to improve the sexism detection under the situation that only about three thousand training data used.
- I have seen the performance improved by learning each individual hidden layer of the BERT transformer encoder layer comparing with only the last output layer, and the benefits of different external features learnt from Perspective API, HurtLex, Empath and Emoji2Vec.
- In the future, different hateful speech dataset in English and other languages can be further investigated to improve the generalization of the model. And if this model can be used for cross-lingual sexism detection, it can be greatly beneficial.