# Report

https://github.com/natalyasegal/ModelCompression_NLP

## Preprocessing

**Stop words.** As observed in the exploratory data analysis (EDA code), the texts contain, on average, 9.81 stop words per sample, corresponding to an average stop-word ratio of 49.22%. Although this indicates that a substantial portion of each text consists of stop words, and they were removed during preprocessing (code) in Part 1 of the project (LSTM, GRU-based models), we did not remove them during preprocessing in this part. For transformer-based models, stop words can contribute to contextual meaning and syntactic structure, and their removal may discard useful information. Therefore, stop words were retained and handled implicitly by the pretrained tokenizer and model during training. Empirically, when we tried Roberta-base with and without stopword removal, we achieved an identical macro F1 of 0.92.

**Tokenization.** Text preprocessing includes the pretrained model tokenizers provided by the Hugging Face Transformers library. Specifically, RoBERTa-based models use a byte-level subword tokenizer (BPE), which converts raw text into subword tokens while preserving punctuation, casing, numbers, and special characters. Tokenization includes inserting special tokens, truncating to a fixed maximum sequence length (maximum 128 tokens), and dynamic padding at batch time. Sequences are padded within each batch to the length of the longest sequence in that batch. Padding tokens are lans are ignored by the model via the attention mask.

**Lemmatization**. We observed that in different classes, some words were predominantly used in specific morphological forms (Fig. 1).

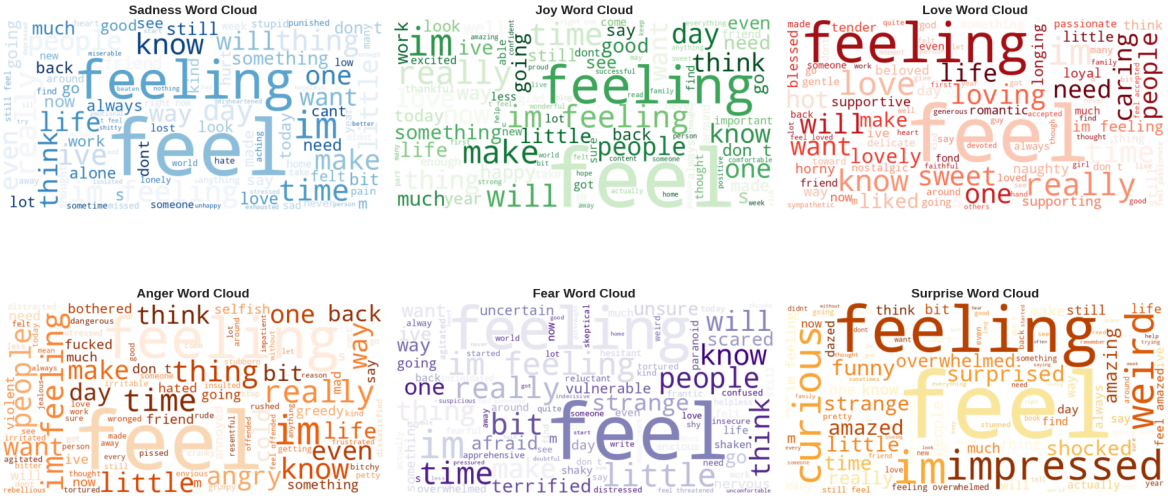


**Fig. 1**

## Metrics

In the validation set, class imbalance persists (see Fig. 2), so metrics suitable for imbalanced multiclass classification should be used for model performance comparison. These include macro F1, balanced accuracy, and Cohen's kappa [1]. Metrics per class are provided as well, but were not used in model selection. See the metrics definitions section below.

## Deciding on the architecture

We have fine-tuned 3 transformer-based models, Bert (code), Roberta (code), and Electra (code) on the train set and evaluated them on the validation set using the Macro F1 metric (see Metrics section). The best-performing model was Roberta, with a Macro F1 of 0.92 (Table 1).

**Table 1.** Comparison of the performance of several Transformer-based models on senses dataset.

| Model name | Macro F1 |
|---|---|
| Roberta-base | 0.92 |
| Bert | 0.91 |
| Electra | 0.91 |
| cardiffnlp_twitter-roberta-base | 0.91 |

## Domain adaptation experiment

We also evaluated a Twitter-domain-pretrained RoBERTa model (*cardiffnlp/twitter-roberta-base*) to assess the impact of domain-specific pretraining on downstream performance. The cardiffnlp/twitter-roberta-base model performs slightly below roberta-base (Table 1, code), suggesting that domain-specific Twitter pretraining does not fully align with the characteristics of the target dataset.

## Model compression (code)

The best-performing model was RoBERTa-base. So we compare different versions of this model (Colab ):

1. Roberta-base
2. BEST_PRUNED - after **pruning**
3. BEST_INT8_CPU - after **quantization**

**Roberta-base.** We initialized the model with publicly available pretrained RoBERTa-base weights and a randomly initialized classification head. The model was then fine-tuned on the downstream task. To mitigate class imbalance during training, we applied class-wise supersampling to underrepresented classes. Class weights were computed after supersampling, so oversampling partially reduced the imbalance and consequently produced less extreme inverse-frequency loss weights.

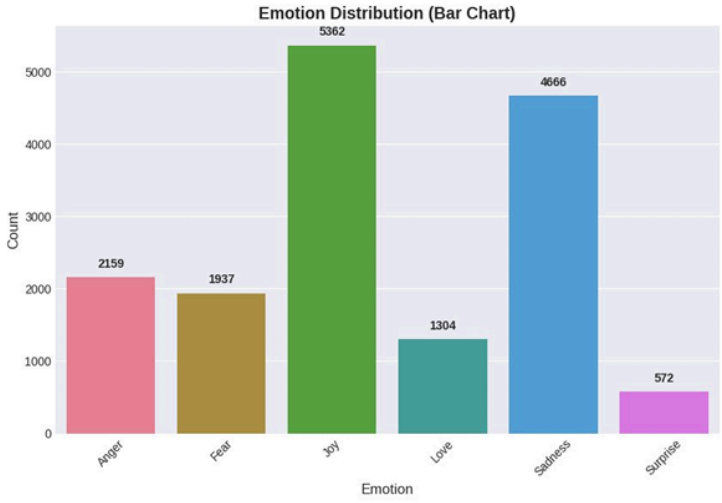


**Fig. 2.** Class imbalance in the training set before super-sampling (see EDA).

**BEST_PRUNED.** The best-performing baseline model was compressed using global unstructured magnitude-based pruning applied to linear layers, followed by a short recovery fine-tuning phase. This procedure resulted in a 6.55× reduction in model size (from 17,656 MB to 2,694 MB) and more than a 2× reduction in CPU inference time per sample (from 17.7 ms to 8.1 ms), while preserving predictive performance. No measurable degradation was observed in Macro F1 or Cohen's κ, and only a minor decrease in balanced accuracy was observed (94.1% → 94.0%). See Table 2-3. We applied global unstructured magnitude pruning to all Linear-layer weight matrices (excluding biases), zeroing the smallest-magnitude weights globally (e.g., 35%), then removed pruning masks to make sparsity permanent and performed a short recovery fine-tuning.

**BEST_INT8_CPU.** The best-performing baseline model was compressed using dynamic post-training INT8 quantization applied to linear layers, starting from the original unpruned model. Quantization substantially reduced model size 17,656 MB → 195 MB (≈90× reduction), and improved CPU inference efficiency (17.70 ms→4.55ms) without additional retraining. Predictive performance was largely preserved, with only a minor reduction observed in Macro F1, Cohen's κ, and balanced accuracy (see Table 2-3).

**Distilroberta-base** achieves inference speed comparable to BEST_PRUNED but exhibits a modest performance drop, reflecting the cost of architectural distillation compared to post-training compression methods.

**Table 2. In the model comparison, BEST_PRUNED and BEST_INT8 denote two model compression techniques.** Performance evaluation was conducted on the CPU. Because differences in the macro F1 and Kappa metrics were minor across models, we retained 3 decimal places.

| Model name | Model Size (MB) | Inference Time PerSample (ms) | **Macro F1** | **Balanced Accuracy** | **Kappa** |
|---|---|---|---|---|---|
| roberta-base | 17655.97 | 17.70 | 0.917 | 94.1% | 0.918 |
| **BEST_PRUNED** | **2693.78** | **8.09** | **0.917** | **94.0%** | **0.918** |
| **BEST_INT8** (quantization) | **194.74** | **4.55** | **0.910** | **93.5%** | **0.910** |
| distilroberta-base | 8818.80 | 8.14 | 0.909 | 93.1% | 0.908 |
| | | | | | |
| cardiffnlp_twitter-roberta-base | 13362.02 | 17.61 | 0.913 | 93.4% | 0.917 |

*Those results can be found: outputs_part2/model_comparison.csv*

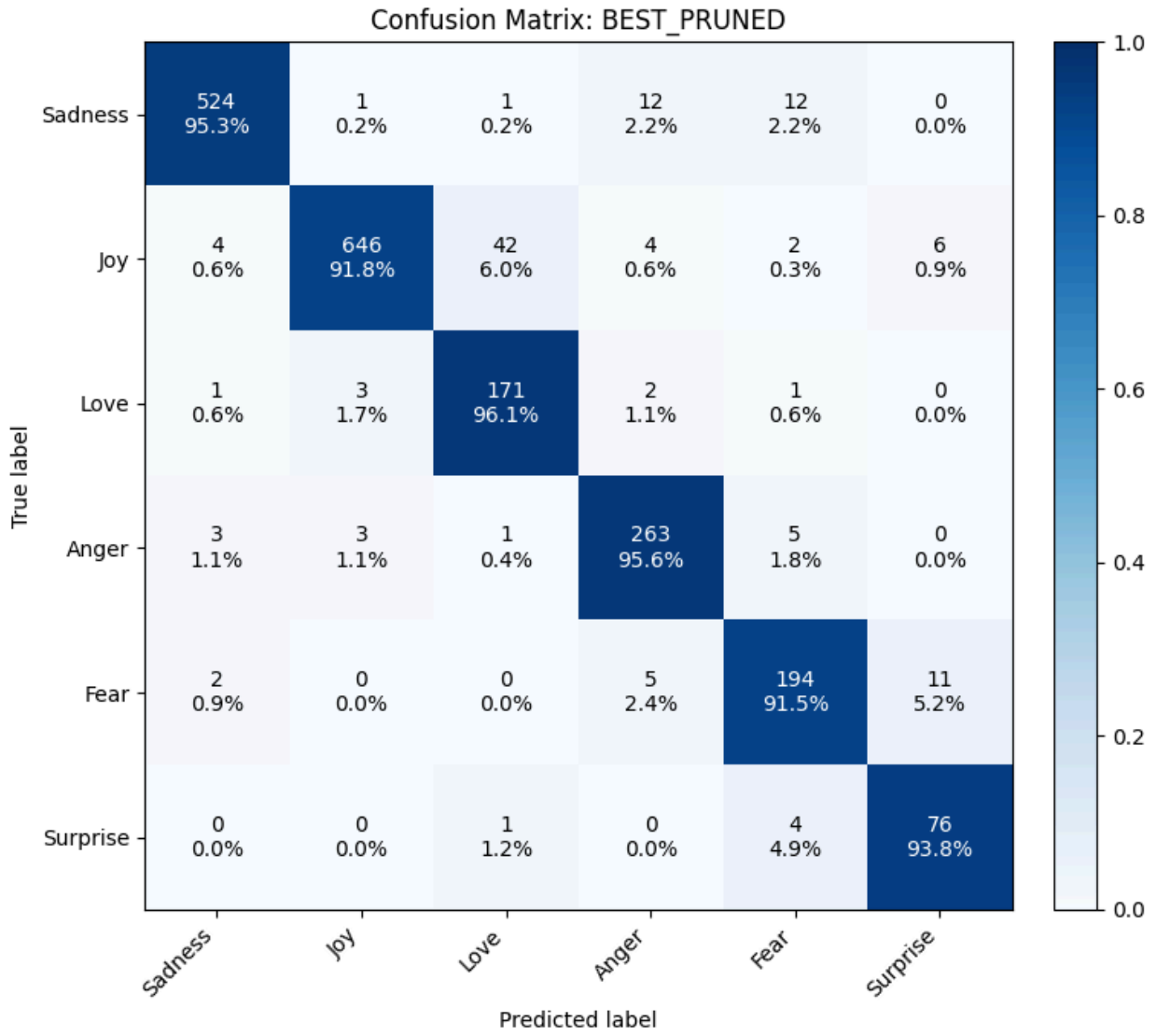


**Fig. 2.** Confusion matrix for **Prunned** version of Roberta's performance on the validation set (see Table 2). Note: the cell color is determined by the percent rather than the value. Created by EvalColab

**Table 3. Per class accuracy (in %) and F1s.**

| Model name | Class 0, Anger | | Class 1, Fear | | Class 2, Joy | | Class 3, Love | | Class 4, Sadness | | Class 5, Surprise | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | F1 | Acc | F1 | Acc % | F1 | Acc % | F1 | Acc % | F1 | Acc % | F1 | Acc % |
| roberta-base | 0.968 | 96.4 | 0.950 | 91.3 | 0.874 | 97.2 | 0.940 | 94.5 | 0.897 | 90.1 | 0.875 | 95.1 |
| **BEST_PRUNED** | 0.967 | 95.3 | 0.952 | 91.8 | 0.868 | 96.1 | 0.938 | 95.6 | 0.902 | 91.5 | 0.874 | 93.8 |
| **BEST_INT8** | 0.960 | 95.1 | 0.945 | 90.8 | 0.869 | 96.6 | 0.941 | 94.9 | 0.880 | 89.6 | 0.869 | 93.8 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (quantization) | | | | | | | | | | | | |
| distilroberta-base | 0.958 | 95.5 | 0.945 | 90.5 | 0.864 | 96.6 | 0.938 | 94.2 | 0.885 | 90.6 | 0.860 | 91.4 |
| | | | | | | | | | | | | |
| cardiffnlp_twitter-roberta-base | 0.961 | 95.1 | 0.945 | 91.8 | 0.879 | 98.3 | 0.945 | 97.5 | 0.889 | 88.7 | 0.852 | 88.9 |

*Those results can be found: outputs_part2/model_comparison.csv*

## Metrics:

Below, we formally define Accuracy, F1, Balanced accuracy, and Cohen's Kappa metrics:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

For K classes:

$$\text{Macro F1} = \frac{1}{K}\sum_{c=1}^{K}\text{F1}_c$$

Where F1 per lass is defined by:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

Here, TP, FP, FN, TN denote the number of true positives, false positives, false negatives, for class c, respectively. All metrics are computed over the validation set.

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$\text{Balanced Accuracy} = \frac{1}{N}\sum_{c=1}^{N}\text{Recall}_c$$

Where $Recall_c$ is the recall per class c

Cohen's kappa (κ) metric is defined as

$K = \frac{P_0 - P_e}{1 - P_e}$ , $P_0$ is accuracy, and is an $P_e$ expected agreement by chance

$$P_e = \sum_i^N \quad p_{true_i} * p_{predicted_i}$$

It quantifies the agreement between predicted and true labels while correcting for agreement expected by chance (Pe). Its values range from -1 (systematic disagreement) to 1 (perfect agreement), with 0 indicating chance-level performance. Unlike accuracy, κ accounts for class imbalance, making it a more robust measure when class distributions are uneven. It is computed by comparing the observed agreement (Po) with the expected agreement (Pe), and normalizing by the maximum possible agreement beyond chance.[1] N denotes the number of classes, $p_{true_i}$ represents the proportion of class i in the true labels, and $p_{predicted_i}$ denotes the proportion of class i in the predicted labels.

## References


1. Viera, A. J. & Garrett, J. M. Understanding interobserver agreement: the kappa statistic. *Fam. Med.* **37**, 360–363 (2005).
2. Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." *arXiv preprint arXiv:1907.11692* (2019).
3. Rokh, Babak, Ali Azarpeyvand, and Alireza Khanteymoori. "A comprehensive survey on model quantization for deep neural networks in image classification." *ACM Transactions on Intelligent Systems and Technology* 14.6 (2023): 1-50.
4. Zhu, Michael, and Suyog Gupta. "To prune, or not to prune: exploring the efficacy of pruning for model compression." *arXiv preprint arXiv:1710.01878* (2017).
5. Jawahar, Ganesh, Benoît Sagot, and Djamé Seddah. "What does BERT learn about the structure of language?." *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*. 2019.


**Table 3:** Model paths (internal use - remove before submission)

| model | path |
|---|---|
| roberta-base | outputs_part2/roberta-base |
| BEST_PRUNED | outputs_part2/BEST_PRUNED |
| cardiffnlp__twitter-roberta-base | outputs_part2/cardiffnlp__twitter-roberta-base |
| BEST_INT8 | outputs_part2/BEST_INT8_CPU |
| distilroberta-base | outputs_part2/distilroberta-base |