Towards Ensemble-based Imbalanced Text Classification using Metric Learning

Takahiro Komamizu

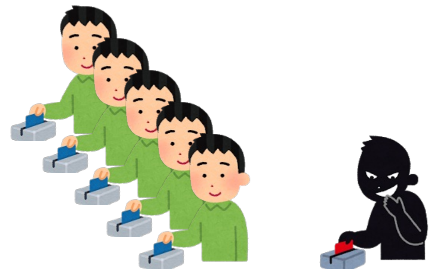
Nagoya University

Japan

This slide is downloadable from
https://taka-coma.pro/pdfs/DEXA2023.pdf

Class Imbalance is Universal Phenomenon

E-mail Spam



Credit Card Fraud

Driving Behavior

- Others in text classification domain
  - the unfair statement prediction in terms of service [17]
  - the hate speech detections [8, 33]

  etc.

Classifiers suffer from Class Imbalance

- Classifiers tend to prefer majority class
  - Choosing majority (say negative) class has more chance to increase accuracy score, beacuse $\#TN \gg \#TP$
    - $accuracy = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN}$
  - Consider 1 positive instance and 99 negative instances
    - All negative: accuracy = 99%
      - For classifiers, it looks (almost) optimal.
- In reality, minority class is more important.
  - What if your spam filter regards all mail as non-spam?
  - What if your fraud detector rageds all as normal action?

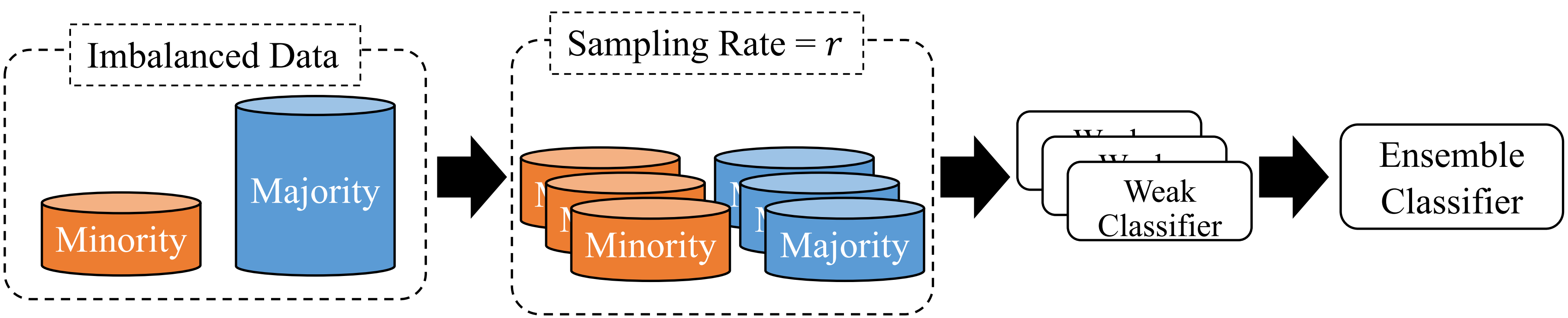
Two Major Approaches for Class Imbalance

- Cost-sensitive learning approach
  - Design cost function that gives higher penalty when classifiers fail to correctly classify the minority classes.
  - Depending on classification methods.
- Data-level approach
  - Add or remove data points so that instances of classes are balanced.
    - Adding: Oversampling / Synthetic oversampling (e.g., SMOTE, SWIM)
    - Removing: Undersampling (US)
  - NOT depending on classification methods.

EasyEnsemble (EE)[18]: ensemble multi samples

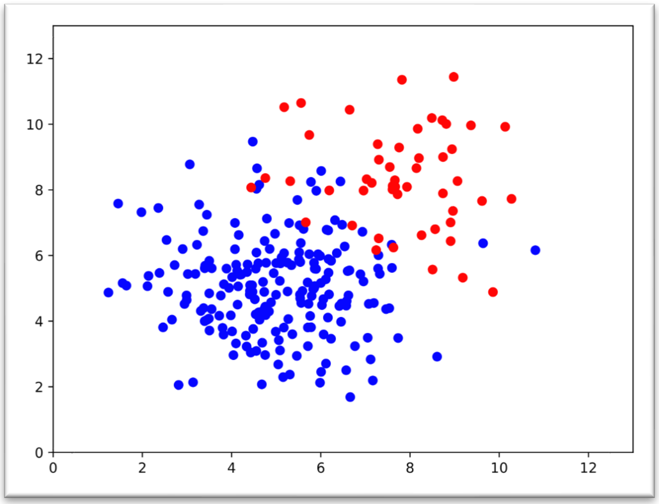
- Simple undersampling wastes major part of samples.

- EE samples multiple times so that most of samples are used in trianing an ensemble classifier.

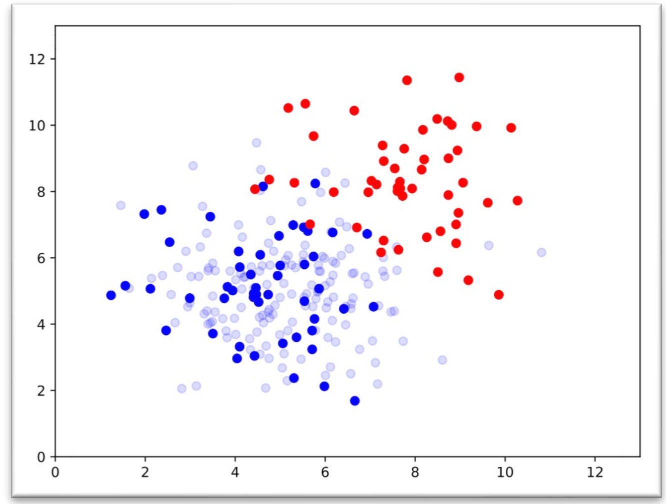


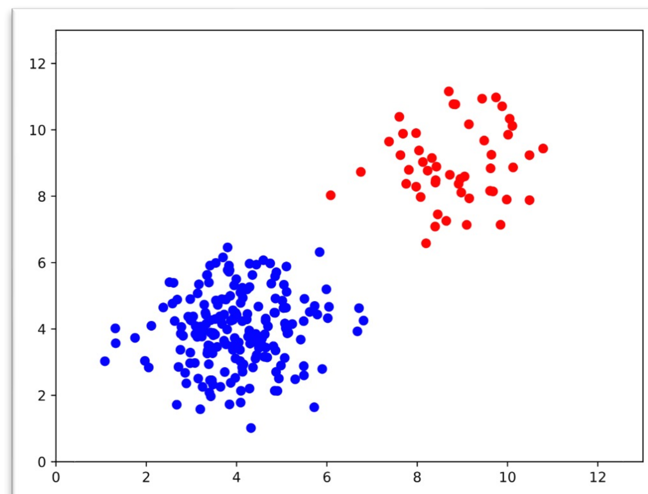
multiple sampling w/ replacement

What about feature space?



US





<u>Metric Learning (ML) e.g., LMNN [19]</u>

Learning a transformation s.t.

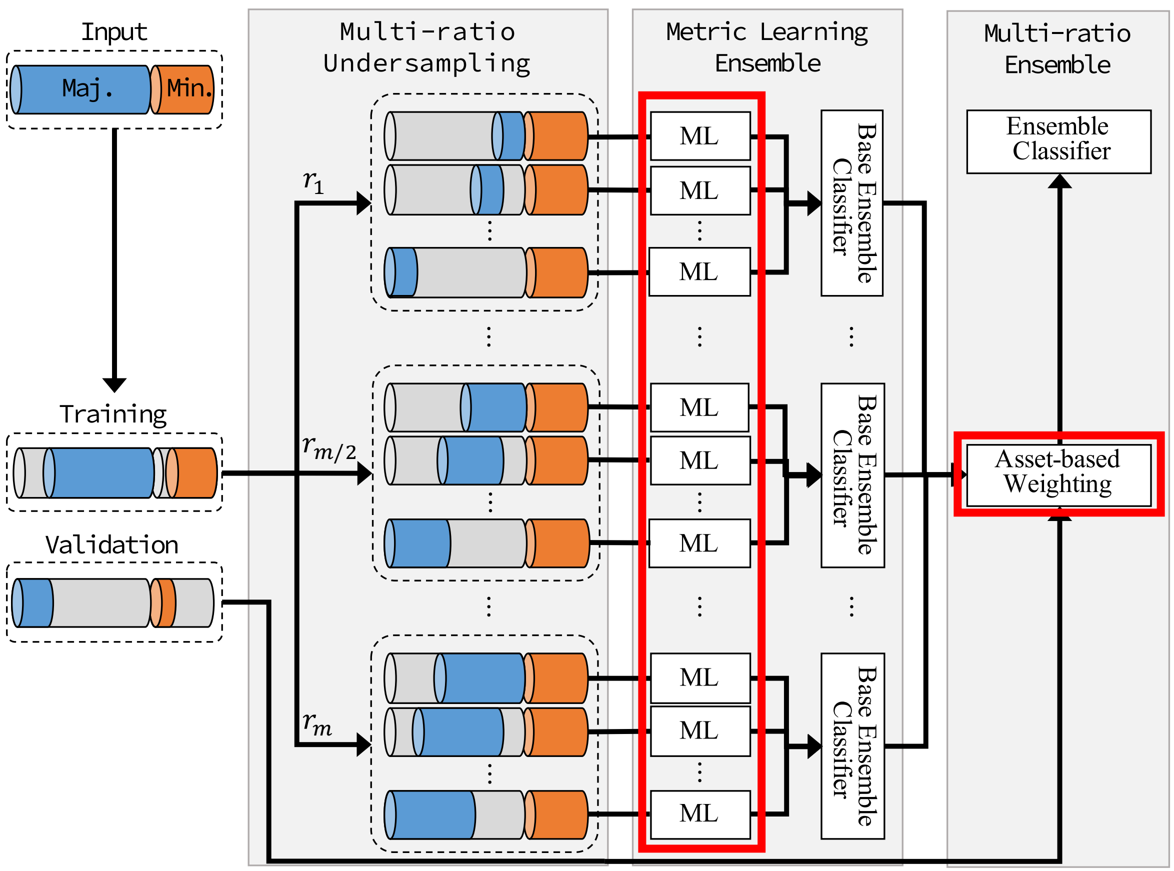
- samples of the same classes get closer,
- samples of the different classes get further

ML also suffers from the class imbalance.

➔ [18] shows US + ML improves classification performance in the class imbalance data.

MMEnsemble[13]: ensemble multiple rates w/ ML

- EE + Multi-ratio US + Metric Learning



Findings

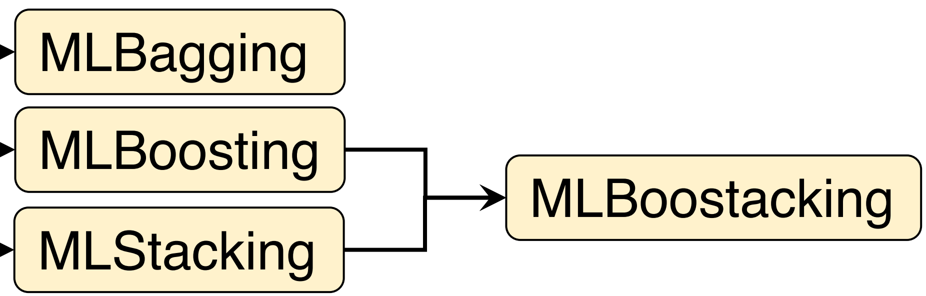
- ML provides positive effects

Limitations

- Learning costs for large number of base classifiers

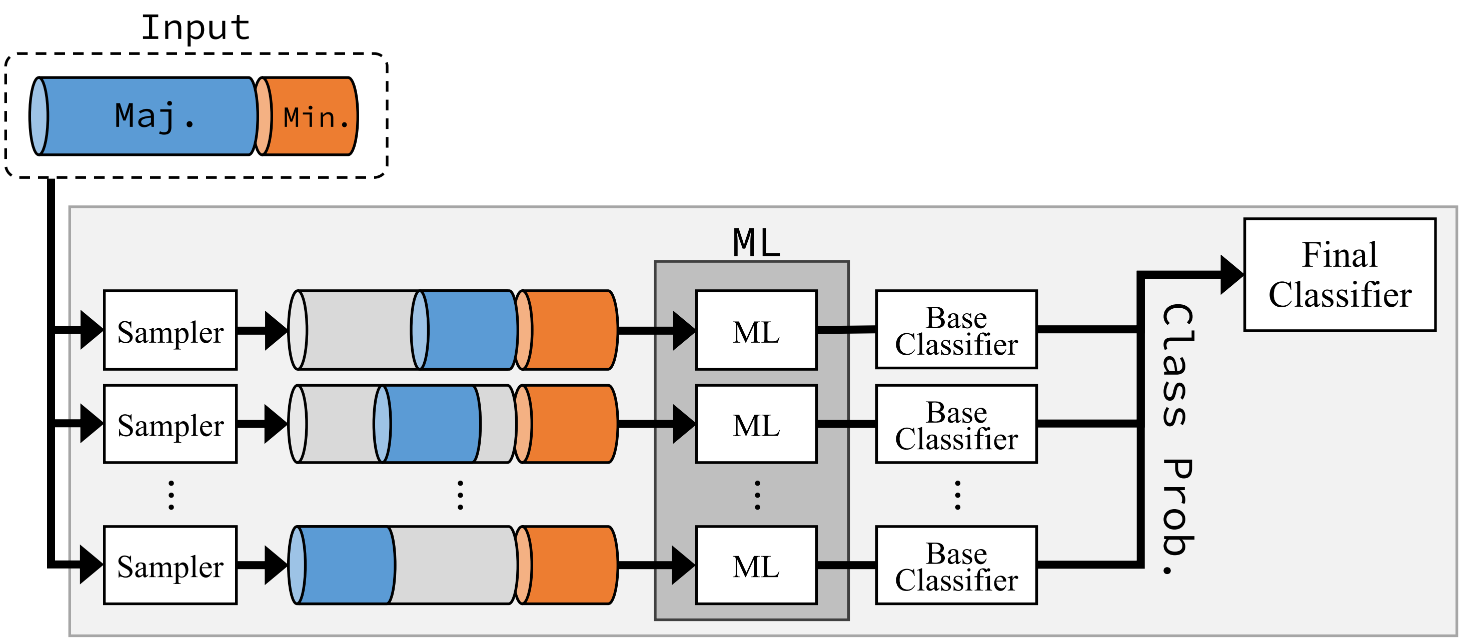
Objective: exploring ensemble schemes for text classification

- Previous approaches: Bagging
- Ensemble schemes
  - Bagging
  - Boosting
  - Stacking



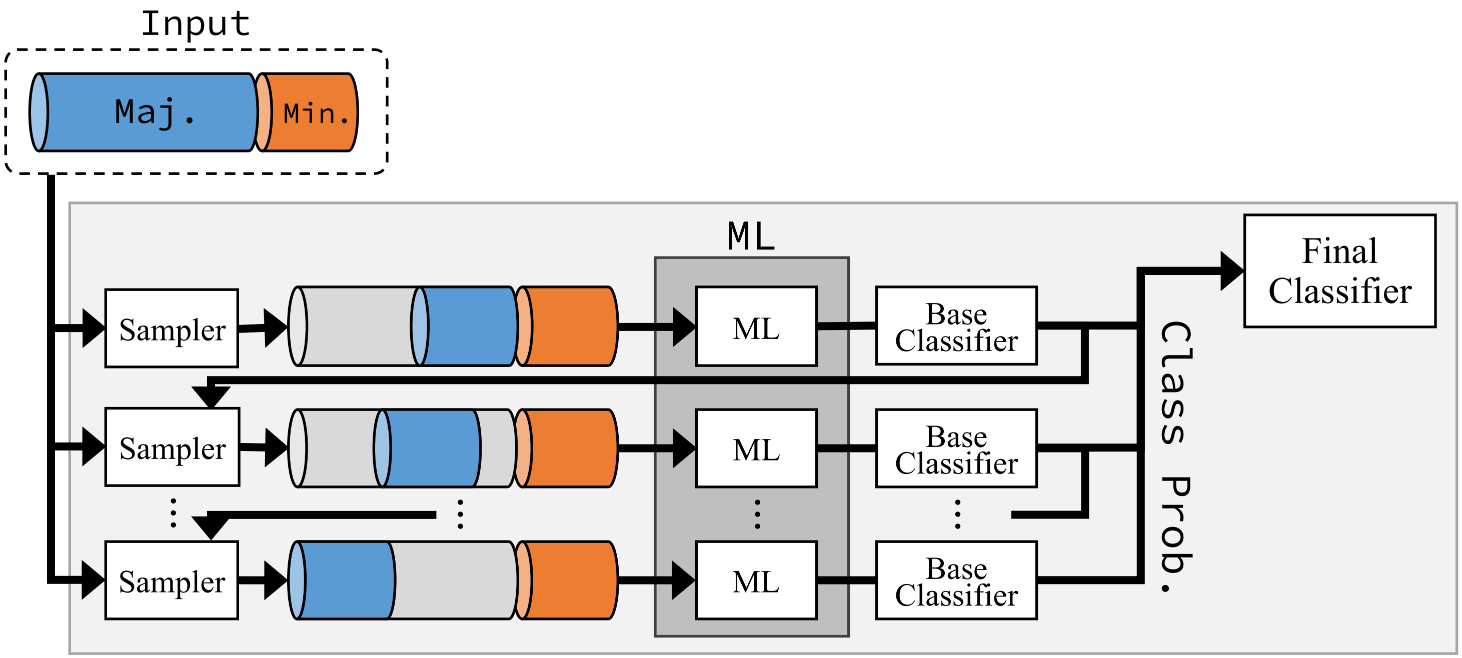
- Text features: NLM-based features
  - Neural language models (NLMs) trained with vast amount of texts
  - In contrast to record data, continuous and high-dims.

MLBagging



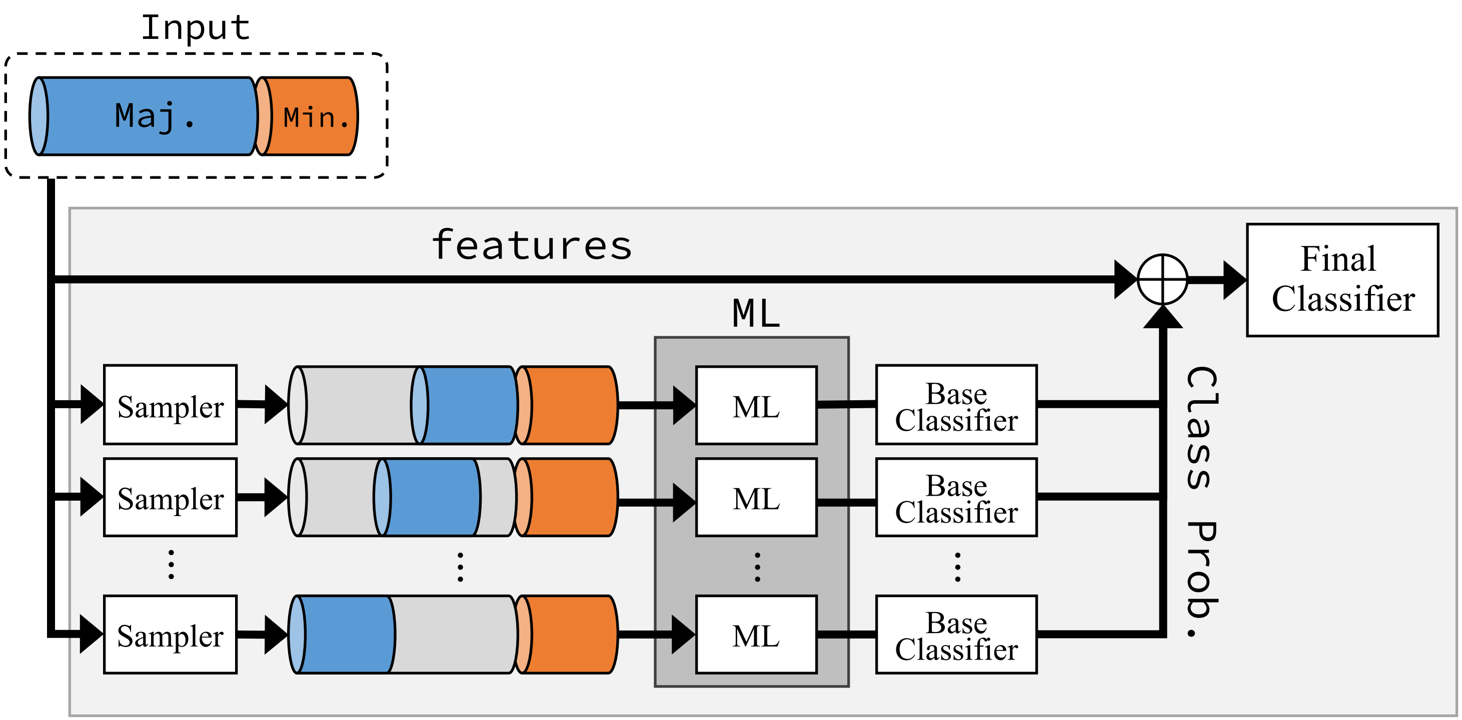
- Independent sampling
- Merge outputs of base classifiers

MLBoosting



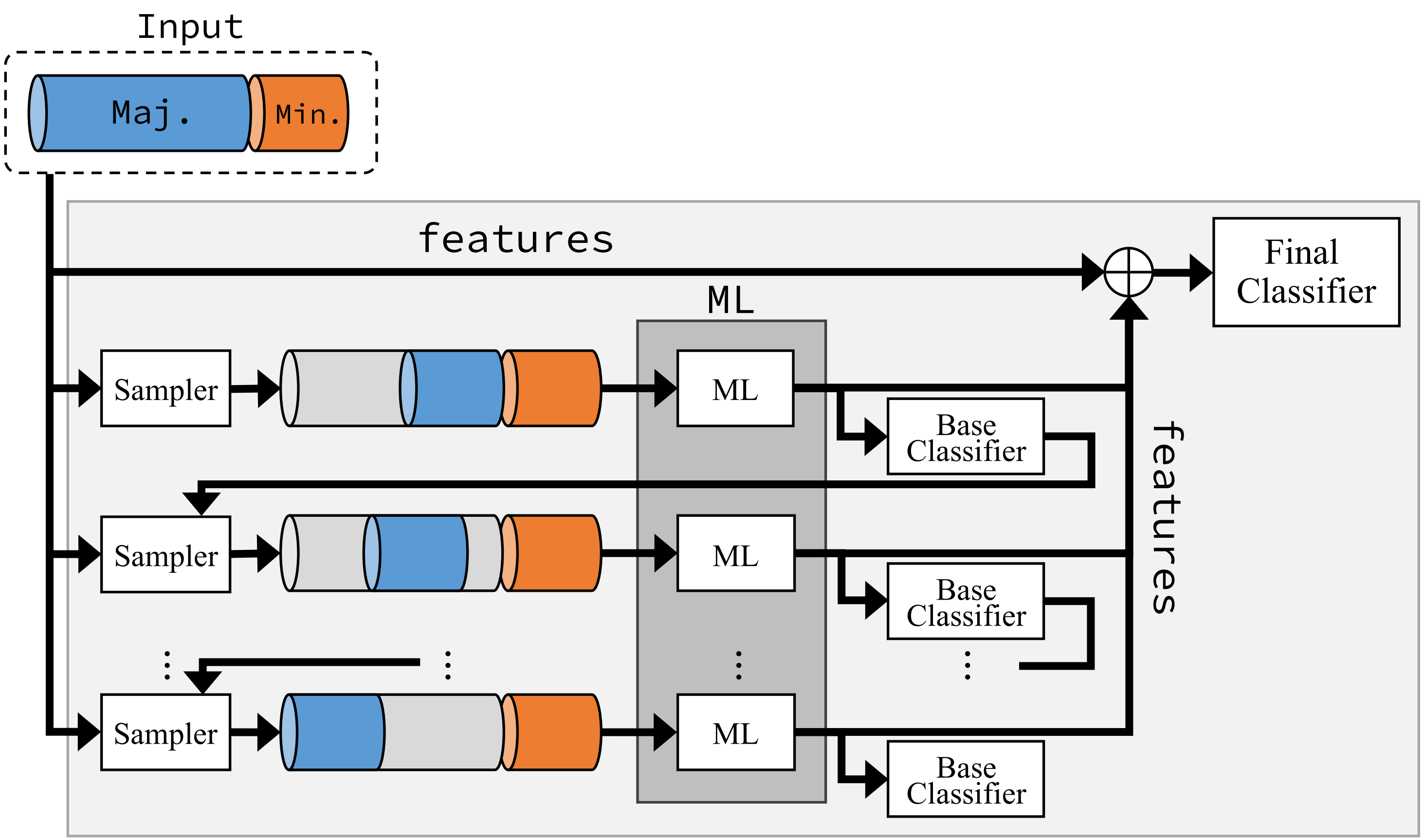
- Sampling based on the previous base classifier
  - To sample harder samples
- Merge outputs of base classifiers

MLStacking



- Probabilities from base classifiers as features
- Combining them with textual features

MLBoostacking



- Boosting + Stacking

Experimental Evaluation

- Research Quesions
  - Are ML-based ensemble methods superior to neural language model-based approaches?
  - Which ensemble scheme is the best?
- Settings
  - Tasks
    - **claudette**: the unfair statement prediction in terms of service [17]
    - **hate-speech18**: the hate speech detection in the Stormfront forum [8]
    - **tweets-hate-speech-detection**: the hate speech detection on Tweets [33]
  - Metrics: Precision, Recall, F_1-score, and Gmean
    - Gmean: geometric mean of recalls on positive and negative classes
  - Base classifier: k-NN classifier (k=5)

Results on **claudette**

Table 2: Comparison for `claudette` dataset.

Model	Feature	Precision	Recall	Gmean	F_1
BERT		.244	**<u>.944</u>**	.754	.382
LegalBERT		<u>.361</u>	.899	.844	.508
LegalBERT+BS		.356	.910	<u>.848</u>	<u>.509</u>
LegalBERT+WCE		.327	.907	.824	.474
LegalBERT+BS+WCE		.338	.931	.842	.492
RUSBoost	LegalBERT	.388	.752	.788	.503
EasyEnsemble	LegalBERT	<u>.451</u>	.840	<u>.844</u>	<u>.579</u>
EasyEnsemble	LegalBERT+Triplet	.432	<u>.854</u>	<u>.844</u>	.565
MLBagging	LegalBERT	**<u>.636</u>**	.894	.910	**<u>.736</u>**
MLBoosting	LegalBERT	.554	.883	.890	.672
MLStacking	LegalBERT	.582	.902	.905	.702
MLBoostacking	LegalBERT	.629	<u>.939</u>	**<u>.919</u>**	**<u>.736</u>**

LegalBERT: pre-trained BERT on the legal domain

BS: balanced sampling, WCE: weighted cross entropy loss

Results on **hate-speech18**

Table 3: Comparison for `hate-speech18` dataset.

Model	Feature	Precision	Recall	Gmean	F_1
BERT		.856	.727	.845	.784
DeBERTa		**.898**	.825	.902	.857
DeBERTa+BS		.890	.885	.934	**.886**
DeBERTa+WCE		.841	.876	.926	.857
DeBERTa+BS+WCE		.791	.916	.942	.847
RUSBoost	DeBERTa	.595	.822	.872	.688
EasyEnsemble	DeBERTa	.670	.921	.932	.775
EasyEnsemble	DeBERTa+Triplet	.683	.937	.941	.790
MLBagging	DeBERTa	.713	.947	.949	.813
MLBoosting	DeBERTa	.724	.957	.956	.824
MLStacking	DeBERTa	.733	.967	.961	.834
MLBoostacking	DeBERTa	.745	**.969**	**.963**	.841

DeBERTa: fine-tuned DeBERTa on the same dataset

BS: balanced sampling, WCE: weighted cross entropy loss

Results on **tweets-hate-speech-detection**

Table 4: Comparison for `tweets-hate-speech-detection` dataset.

Model	Feature	Precision	Recall	Gmean	F_1
BERT		.780	.730	.847	.752
DiRoBERTa		**.847**	.547	.733	.655
DiRoBERTa+BS		.718	.704	.827	.700
DiRoBERTa+WCE		.712	.595	.757	.634
DiRoBERTa+BS+WCE		.485	.840	.882	.607
RUSBoost	DiRoBERTa	.547	.840	.889	.660
EasyEnsemble	DiRoBERTa	.647	.959	.961	.776
EasyEnsemble	DiRoBERTa+Triplet	.658	.964	.963	.782
MLBagging	DiRoBERTa	.704	.958	.964	.812
MLBoosting	DiRoBERTa	.625	.864	.910	.723
MLStacking	DiRoBERTa	.673	**.967**	.966	.794
MLBoostacking	DiRoBERTa	.722	.964	**.967**	**.823**

DiRoBERTa: fine-tuned distilled RoBERTa on the same dataset

BS: balanced sampling, WCE: weighted cross entropy loss

Lessons Learned

- Q1. Are ML-based ensemble methods superior to neural language model (NLM)-based approaches?
  - Yes, esp. in Recall and Gmean metrics.
  - Superior to learned representations via a deep metric learning, Triplet loss.
- Q2. Which ensemble scheme is the best?
  - MLBoostacking: Boosting + Stacking
    - Stacking features from ML to the final classifier was effective.

Conclusion

- A serise of ensemble approaches using metric learning to deal with the class imbalance issue in text classification.
- NLM-based approaches were not enough to learn the classifiers. So, more sophisticated representation learning is necessary in the text classification problem.
  - Since NLMs are not designed for any specific natural language processing task, to apply them into some task, sophisticated approaches are still needed.