arXiv:submit/5314789 [cs.AI] 24 Dec 2023

Active Learning for Question Difficulty Prediction*

Shashwat Gupta[1][0009−0003−8037−2348], Jibril Frej[2], Paola Mejia[3], and Tanja Kaesar[2]

[1] IIT Kanpur (Indian Institute of Technology - Kanpur), India
shashwatg20@iitk.ac.in, guptashashwatme@gmail.com
[2] EPFL (Ecole Polytechnique Federale de Lausanne) jibril.frej@epfl.ch
[3] EPFL (Ecole Polytechnique Federale de Lausanne) paola.mejia@epfl.ch
[4] EPFL (Ecole Polytechniqe Federale de Lausanne) tanja.käser@epfl.ch

Abstract. This paper focuses on question difficulty estimation (calibration), and its applications in educational scenarios and beyond. The emphasis is on the use of Active Learning to bound the minimum number of labelled samples that we need. It also explores using various SOTA methods for predicting question difficulty, with a specific focus on German textual questions using the Lernnavi dataset. The study refines preprocessing techniques for question data and metadata to improve question difficulty estimation.

Keywords: Active Learning · Text Classification · Question Difficulty Caliberation · Natural Language Preprocessing (NLP)

1 Introduction

Question Difficulty Estimation (or 'calibration') is quite useful for educational Scenarios. The most common use is to design adaptive Tests (next question based on response and difficulty of the previous question) and fair test systems (balanced composition of difficulty across sets).

The most generic use of the approaches we develop here is for any task that needs the classification of textual sequences into ordinal classes. The most trivial examples include the cases with 3 ordinal classes e.g., Twitter sentiment analysis (positive, negative, and neutral), customer support ticket prioritisation (low, medium, or high), resume shortlisting (low fit, medium fit, high fit), etc. More advanced applications might include: essay grading (A, B, C, D, E, F grades) and severity classification in health-care (mild, moderate, severe, critical)

Recently, the work by Benedetto et al. [1], explores using transformers for this task. We further take a step ahead in utilising various state-of-the-art methods (including transformers) for question difficulty estimation, parsed with our own preprocessing for German text. On the best models and preprocessing, we use Active Learning methods to improve the model's performance. The benefits of

* Supported by organization ML4ED Lab, EPFL.

Active Learning include learning on smaller-set, reduced computational costs and better learning by less overfitting.

Our main contributions include: (1) Exploring and finding methods to predict question difficulty, especially for German textual questions for our proprietary dataset - Lernnavi, (2) Exploring and refining techniques to preprocess question data and meta-data to question difficulty estimation using popular text classification methods, (3) Deploy Active Learning to find out the minimum number of labelled instances needed and the strategies to use them. (4) Identify research gaps to guide researchers in this direction further. We track the Balanced Accuracy ROC-AUC score and confusion matrix - the common metrics for the imbalance dataset (and sometimes report the losses and Accuracy if necessary).

Our code can be found here: `https://github.com/epfl-ml4ed/question-difficulty.git`

2 Related Work

To come up with research work, we searched for keywords for each of the above-mentioned contributions in top conferences such as Nature, EMNLP, and Google Scholar and read the top results from the papers. We consulted the website paperswithcode [2] for the benchmarks, which lists different datasets with various benchmarks on them. Unlike, [3], our approach was to try both – relevant papers with the task at hand and also the top approaches/papers that excelled in a component of the task/objective.

2.1 Question Difficulty

As mentioned in [3], there are 3 popular definitions of question difficulty:

Classical Test Theory According to CTT, Observed Score (X) is approximately equal to true score (T) with an error term (E). True Score (T) is the expected correctness of an infinitely long run of repeated independent test administrations.

$$X = T + E$$

We make the following assumptions: We make the following assumptions:

1. $Corr(T, E) = 0$
2. $E \sim \mathcal{N}(0, \sigma^2)$
3. E from different tests are uncorrelated

The difficulty is expressed using the p-value, a continuous value ranging from 0 to 1. The p-value represents the fraction of correct responses in the population, with higher values indicating easier items. One advantage of CTT is its simplicity in computation and understanding. A limitation of CTT is that it does not consider students' skill levels when estimating item difficulty, relying solely on the fraction of students who answer a question incorrectly.

Item Test Theory IRT [5] connects latent traits to students and questions. In its simplest form, the Rasch Model [6] pairs a skill level θ to each student and a difficulty level b to each question. IRT's key feature is "invariance": item traits are independent of test takers' ability distribution, and a question's difficulty is consistent irrespective of students' skills. IRT assumes that (i) individuals and (ii) their responses are independent. For question, j with trait b_j, the item response function (i.r.f.) calculates the probability (PC) that the student i with skill level θ_i answers correctly. The i.r.f. Formula is:

$$PC = \frac{1}{1 + e^{-1.7 \cdot (\theta_i - b_j)}}$$

(1) where 1.7 was found to yield accurate results. If a question is too hard or easy (i.e., $b_j \to \infty$ or $b_j \to -\infty$), all students answer similarly (i.e., $PC \to 0$ or $PC \to 1$). Hence, avoiding overly easy or hard questions is crucial. IRT yields real-valued difficulties, sometimes discretized for practicality.

Manual Definitions We assume bias in our subjective definitions is minimized, given that annotators are experts in their fields. If a factor greatly impacts difficulty, our objective function can capture it, with model overfitting and subjectivity-induced noise kept at bay.

We categorize calibration questions into two types:

1. Language Assessment (LA): LA can further be classified as Comprehension and Knowledge.
2. Current Knowledge Assessment (CKA)

Benedetto et al. [1] excel in Question Difficulty Estimation compared to traditional Manual Calibration and Pretesting methods by employing transformers like BERT and DistilBERT. We forgo wrongness, p-value, and IRT metrics, focusing solely on textual information to estimate question difficulty. The benefit of human annotation is its comprehensive capture of question difficulty factors. However, bias is an inherent limitation. We work exclusively with LA questions, dismissing CKA ones. LA approaches often utilize theoretically backed metrics like readability formulas and word complexity measures, while CKA leverages learned features. As empirical evidence suggests, NLP techniques tend to perform better on LA tasks.

2.2 Metrics and Imbalance

Imbalanced data can lead to model biases. It can learn only the majority class and thus be as good as a random model. Our approach to mitigating this, influenced by the blog [38], involves selecting suitable metrics, implementing a robust cross-validation strategy, and weighting classes to favour the minority.

Metrics that help identify data imbalance include (ordered by their decreasing sensitivity to imbalance or increasing sensitivity to balance) **red**define

1. visual inspection,
2. Shanon entropy [39] (most reliable and sensitive to imbalance)
3. minimax ratio
4. and an imbalance calculation.

Regarding the metrics, accuracy is notorious for imbalanced classification. The popular metrics include :

1. F1 Score: harmonic mean of precision and recall
2. Indexed Balanced Accuracy : $IBA = (1 + \alpha * D) * (GM)$
3. Dominance : D = Recall - Precision
4. Geometric Mean (GM) : $GM = \sqrt{Precision * recall}$
5. MCC: correlation coefficient between the observed and predicted binary classifications
6. ROC-AUC (Receiver Operating Characteristic) Curve: TPR(i) vs FPR(i) overall threshold values

We chose (Indexed) Balanced Accuracy and ROC-AUC Score for this study. However, we do report other metrics when necessary.

For dealing with imbalanced classification, popular methods for data augmentation include: 1. Undersampling the abundant class can be achieved through random undersampling, SMOTE + undersampling [40], near sampling, cluster centroids, and Tomek links. 2. Modifying cost functions to penalize misclassification of the rare class more, such as by tweaking an SVM or using the Focal loss function. 3. Hybrid approaches like downsampling and unweighting the majority class, ensemble models, or clustering abundant classes.

Since certain models, namely Distilbert and HAHNN were not much affected by imbalance, and the imbalance was moderate (by Shanon Entropy), we didn't heavily explore imbalance handling through data augmentation.

2.3 Active Learning

In this section, we provide a holistic view of Active Learning and, later on, dive into some selected SOTA techniques for Query Strategy. The active learner combines a model, a query strategy and (optionally) a stopping criterion. The overall objective is to minimise the interactions between the oracle and the active learner.

The Steps of the overall process are as follows:

1. Oracle requests the Active Learner for unlabelled instances
2. Active learner passes the batch of unlabelled instances based on the query strategy
3. Oracle labels the instances and updates the active learner. After each update, the model is trained.
4. Repeat the above steps until the stopping criterion is met.

There are usually 3 scenarios [7] to apply active learning :

1. Pool-based: the learner has access to the closed set of unlabelled instances, called the pool; usually done in batch-mode
2. Stream-based: learner receives a set of instances at a time and has options to keep or discard it
3. Membership query synthesis, in which the learner creates the new artificial instances to be labelled.

Despite Deep Learning being the new buzzword, Neural Networks are not more prevalent in Active Learning applications [9]. This is because of the following 2 factors:

1. Uncertainty in Neural Networks: Neural Networks do not provide accurate estimations of uncertainty; thus, the trivial technique of uncertainty sampling, and other techniques, are not straightforward to apply. Though there have been attempts to solve this issue by ensembling, learning error estimates, using Bayesian extensions, obtaining uncertainty estimations using dropouts, use probabilistic NNs to estimate predictive uncertainty. However, Bayesian and ensemble approaches quickly become infeasible on larger datasets, and NN is generally overconfident in their predictions.

2. Contrasting Paradigms DL techniques are known to overfit small datasets (typical use-cases of Active Learning), and are computationally expensive (training and hyperparameter-tuning stages). However, small datasets issue is handled mostly through pre-training or transfer learning. Hyperparameter tuning is not done explicitly; rather, results of related works are used. The Deep Learning field inspires the recent interest in Active Learning. The recent developments like architectures, embeddings, transfer learning, and sub-word operations by LMs for out-of-vocabulary words make augmenting AL with DL an exciting area of research.

Query strategies Since one of our main objectives for us is to determine how many labelled samples we need, we discuss at length the query strategies that we use on top of good models for text/ordinal classification. We follow the classification done by [10]. (other classifications, such as those by Aggarwal et al and Fu et al., are also similar).

1. Random: used as baselines, with competitive performance when labelled pool grows larger
2. Informativeness: strategies mostly assign an informative measure to each unlabeled instance individually
3. Representativeness: Only considering the informativeness of individual instances may have the drawback of sampling bias, and the selection of outliers. Therefore, representativeness, which measures how instances correlate with each other, is another major factor to consider when designing AL query strategies.
4. Hybrid: We can combine informativeness, and responsiveness can be combined, for instance, querying, leading to hybrid strategies

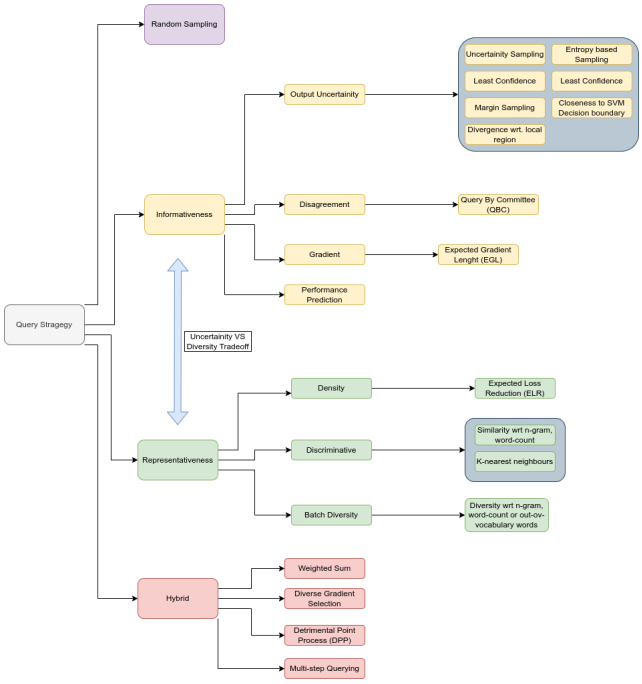


Fig. 1: Active Learning Query Strategies with Zhang's classification

3 Dataset Description

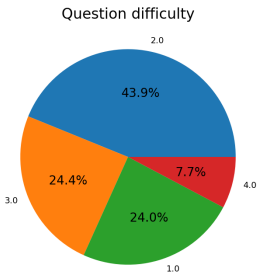


Fig. 2: 1 : very easy, 2 : easy, 3 : moderate, 4 : hard

We describe the Lernnavi Dataset, a proprietary dataset useful for learning German and Mathematics. Since we used only the German textual questions, we described the German features. Data description is useful for the following reasons:

1. The statistics provide a glimpse of data (we can't release the full proprietary data) and provide a tentative idea of a generic setting where the results of this study might be useful
2. The statistics give us a direction of important features and metrics to try out.

3.1 Class Distribution and Imbalance

We find that close to half of the questions are easy and very easy, and moderate are close to a quarter. The hard questions are in the minority (just 7.7%). The pie-chart shows a moderate imbalance. This is confirmed by imbalance statistics (defined in Related Work):

1. Shannon Entropy: 0.8990
2. Range Imbalance: 0.6380
3. Minimax Ratio : 0.1766 (approx. = $\frac{1}{5}$)

Because of moderate imbalance, and decent performance on models, we chose imbalance-immune metrics and did not use feature-augmentation methods.

3.2 Relation with other features

We used an Extra-tree classifier to calculate feature importance for non-numerical features. The data is unwounded and then converted to one-hot encoding. Then an instance of the Extra Trees Classifier fits the model to the encoded features and target and retrieves the feature importance.

For numerical features, we used the Spearman correlation coefficient.

	col_name	Importance
0	type	42.925669
1	description	16.357530
2	title	11.333136
3	ArticleTitle	11.253393
4	solutionSteps	6.164109
5	clozeText	4.906999
6	wrong	3.719312
7	correct	3.339853
8	TopicName	0.000000

Fig. 3: Feature Importance using Extra-tree classifier

	DFQ_label	estimatedDuration	Words
DFQ_label	1.000000	0.422517	0.263460
estimatedDuration	0.422517	1.000000	0.266454
Words	0.263460	0.266454	1.000000

Fig. 4: Spearman correlation matrix for numerical features

4 Preprocessing

We converted 8/12 of the types of German Questions to the Generic Question Format (explained in Fig. 4) and preprocessed text based on data description as follows:

1. Empirical Form (which we found to be good empirically)
2. Frequency order (high frequency means that a particular substring occurs a lot of times in the dataset, where each substring is a cell entry in data frame)
3. Importance order (explained above)

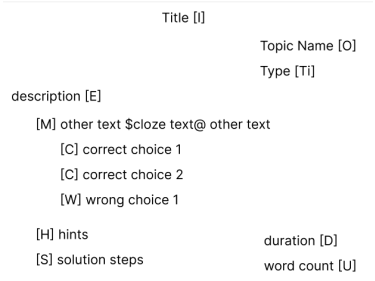


Fig. 5: Generic Question Format with tags in []

Since LMs like BERT truncate the text based on length, we found it useful to use 2 forms: low to high and high to low for the above heuristics. We also tried to experiment with the non-tagged version. Following are the preprocessing that we came up with:

1. description in middle with surrounded by high-frequency elements.
2. low to high-frequency elements
3. high to low-frequency elements
4. plain-text version of preprocess_1
5. plain-text version of preprocess_2
6. plain-text version of preprocess_3
7. high to low importance elements
8. plain-text version of preprocess_7

uniqueId	0.0579Qg9tlXJa7fVwD34WlCn
preprocess_1	[T1] [H] [S] [W] genitivobjekt [C] präpositionalgruppe [W] adverbiale [O] [I] SATZGLFORM Kopie [D] 60.0 [E] Nicolas arbeitete stundenlang an seiner Zeichnung für das Projekt im Fach Bildnerisches Gestalten.
preprocess_2	[D] 60.0[T1] [O] [I] SATZGLFORM Kopie [E] Nicolas arbeitete stundenlang an seiner Zeichnung für das Projekt im Fach Bildnerisches Gestalten. [H] [S] [W] genitivobjekt [W] adverbiale [C] präpositionalgruppe
preprocess_3	[C] präpositionalgruppe [W] genitivobjekt [W] adverbiale [S] [H] [E] Nicolas arbeitete stundenlang an seiner Zeichnung für das Projekt im Fach Bildnerisches Gestalten. [I] SATZGLFORM Kopie [O] [T1] [D] 60.0
preprocess_7	[O] [T1] [E] Nicolas arbeitete stundenlang an seiner Zeichnung für das Projekt im Fach Bildnerisches Gestalten. [I] SATZGLFORM Kopie [S] [H] [W] genitivobjekt [W] adverbiale [C] präpositionalgruppe [D] 60.0
preprocess_4	genitivobjekt prapositionalgruppe adverbiale satzglform kopie 600 nicolas arbeitete stundenlang an seiner zeichnung fur das projekt im fach bildnerisches gestalten
preprocess_5	600 satzglform kopie nicolas arbeitete stundenlang an seiner zeichnung fur das projekt im fach bildnerisches gestalten genitivobjekt adverbiale prapositionalgruppe
preprocess_6	prapositionalgruppe genitivobjekt adverbiale nicolas arbeitete stundenlang an seiner zeichnung fur das projekt im fach bildnerisches gestalten satzglform kopie 600
preprocess_8	nicolas arbeitete stundenlang an seiner zeichnung fur das projekt im fach bildnerisches gestalten satzglform kopie genitivobjekt adverbiale prapositionalgruppe 600

Fig. 6: An example consisting of various preprocessing that we did.

5 Models and Experimental Setup

We describe SOTA models for Text classification to get an idea of the baselines and accuracy results, and then select the best model and preprocessing to run various Active Learning strategies.

5.1 Normal Text Classification Methods

We use V100 GPU on Google Colab for our experiments with the following models.

5.1.1 Distilbert (distilbert-base-german-cased) BERT [19] is a pre-trained language model (LM) that reached state-of-the-art performance in many language tasks. Its key technical innovation was the application of the Transformer, a popular self-attention model [20], to language modelling. BERT is originally trained to address two tasks: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). However, we can stack an additional layer on top of the original network and then retrain it on the desired task ('finetuning'), which is less computationally expensive. BERT is a large model and therefore requires many resources for training and fine-tuning. DistilBERT [21] is obtained by distilling BERT and retains 95% of BERT's performance on several language understanding tasks using about half the number of parameters of BERT. Knowledge distillation is a compression technique in which a small model is trained to reproduce the full output distribution of a larger model [22].

We consider Distilbert pre-trained on German data. The rationale for using the model was because of its MLM nature. By the theory of representativeness in predicting question difficulty in grammatical tasks [1,3], the more repeated the information, the easier it is to answer the question, especially the grammatical ones. This relates to the ease of predicting the next token by MLM models. Besides, the transformer architecture is really amazing at capturing the dependencies between words in a sentence. Benedetto et al. [1] also used BERT and DistilBERT for the task, but he used it for English text. The fact that the model was fine-tuned in German for unsupervised learning tasks made it even more of a choice to consider for our experiment. The 512 max_length of input sequence, enables us to see the roles of selectively using low and high-frequency features on difficulty estimation.

Experimental Setup: We run the experiment for all preprocessing and take the values obtained. For 25 epochs, we find that the values of Accuracy, and Balanced Accuracy are approximately similar. The loss used is a weighted combination of cross-entropy loss and KL divergence.

5.1.2 HAHNN Hierarchial Attentional Hybrid Neural Network (HAHNN) [23] is the SOTA model for Yelp-5, a dataset with ordinal classes similar to ours. It even outperformed other models on IMDB Movie Reviews Dataset. HAHNN leverages a hierarchical architecture and attention mechanisms. This system first decomposes input data into several levels of abstraction, each processed by a distinct subnetwork layer in the hierarchy. The attention mechanism in each layer selects and prioritizes salient information, enhancing interpretability and model performance. The hybrid nature of the model signifies the combination of different neural network types (like CNNs, RNNs, etc.) at various levels of the hierarchy, allowing it to handle a diverse range of data complexities and structures effectively.

Experimental Setup: We run HAHNN with the following parameters for each of the preprocessing by running over each preprocessing (with LSTM and GRU) 5 times and taking the average.:

1. We generate FastText (by Meta AI) from scratch.
2. RNN type : GRU and LSTM
3. Learning rate = 0.001
4. Epochs = 20 (statistics stabilised beyond this)

5.1.3 XLNet XLNet (eXtreme-Learning Network) [24] is a SOTA model for Amazon 5 dataset (a dataset for multi-class classification) and outperforms BERT on 20 NLP tasks. It is based on Transformer-XL architecture, which employs a two-stream self-attention mechanism.

BERT corrupts input with masking, suffers from pretrain-finetune discrepancy (real-life data has no masked inputs), and neglects dependency between masked positions. XLNet incorporates the following features to deal with these issues of BERT:

1. XLNet uses autoregressive objective (unlike MLM by BERT) and Permutation-based Training to better capture dependencies between all tokens in the input sequence.
2. Relative Positional Encoding to capture the relationships between tokens without relying on absolute positions.

Experimental Setup: Due to memory constraints, we used the 98 percentile of the length for the threshold. Length_threshold = 1778 (when max=3124), approximately halving the max_length.

5.1.4 Electra ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements [25] Accurately) introduces a new pre-training approach that differs from standard masked language modelling objectives used by models like BERT. ELECTRA utilizes a discriminator model that is trained to distinguish "real" tokens in the input from "fake" ones that have been replaced by a generator model. Because it predicts every token rather than just masked ones, ELECTRA can learn more efficiently than BERT, as it makes better use of the computation in the pre-training phase. This leads to ELECTRA models performing similarly to BERT but with a fraction of the computational resources.

5.2 Active Learning Methods

We use the popular Active Learning methods to determine Question Difficulty. We mainly follow the fundamental approaches that are commonly used. The Small-text library [26] has lots of methods implemented in it. We use some of those to come up with various solutions.

The initial exploratory methods are helpful for us because there is less literature on traditional AL methods with DNNs as models, as mentioned in [30].

Thus, we aim to obtain baseline results from the Small-text library using these methods.

Note: Notation-wise, we denote instances by x_1, x_2, . . . , x_n, the number of classes by c, the respective label for instance x_i by y_i (where $\forall i : y_i \in \{1, ..., c\}$), and $P(y_i \| x_i)$ is a probability-like predicted class distribution.

5.2.1 Random Sampling (above-described method without active learning was random sampling. We use Random Sampling to provide us with a baseline to compare the performance of various methodologies.

5.2.2 LeastConfidence Least Confidence [27] selects instances whose most likely label has the least confidence according to the current model:

$$\arg\max_{x_i}[1 - P(y_i = k_1^*|x_i)]$$

5.2.3 PredictionEntropy PE [28] selects instances with the highest entropy in the predicted label distribution with the aim of reducing overall entropy:

$$\arg\max_{x_i}\left[-\sum_{j=1}^{c} P(y_i = j|x_i) log P(y_i = j|x_i)\right]$$

5.2.4 BreakingTies/Margin Sampling Breaking Ties [29] takes instances with the minimum margin between the top two most likely probabilities:

$$\arg\max_{x_i}[P(y_i = k_1^*|x_i) - P(y_i = k_2^*|x_i)]$$

; where $k_1^\star$ is the most likely label in the posterior class distribution $P(y_i|x_i)$, and k_2^* the second most likely label respectively. In the binary case, this margin is small if the label entropy is high, which is why BT and PE then select the same instances.

5.2.5 EmbeddingKMeans EmbeddingKMeans [31], is an innovative way to address the 'cold-start' problem in active learning, which is the challenge of selecting data samples for initial training when no labelled data is available. Using self-supervised learning (k-means clustering), it leverages the abundant unlabeled data to initialize the active learning process rather than starting from a few hand-labelled samples.

5.2.6 GreedyCoreset Coreset [32] is a compact representation of data such that models trained on coresets, are provably competitive with models trained on the full data set. The core set is constructed by iteratively selecting instances that are both uncertain and representative of different regions of the input space.

To achieve this, the authors leverage a technique called k-means clustering. Initially, a small labelled dataset is used to train a CNN model. Then, the remaining unlabeled data is clustered using k-means, and the most representative instances from each cluster are selected for labelling. These newly labelled instances are added to the training set, and the process is repeated iteratively.

5.2.7 BALD (time-taking)

BALD (Bayesian Active Learning by disagreement) [33] is a Bayesian active learning framework that allows for efficient data selection in both classification and preference learning tasks. They present algorithms that leverage Bayesian models to estimate the uncertainty of the model's predictions and select the most informative instances for labelling. In the classification setting, the framework proposes two different active learning algorithms. The first algorithm selects instances based on the expected reduction in uncertainty, while the second algorithm incorporates diversity by considering the representation of different regions of the input space.

5.2.8 LightweightCoreset Coresets are compact representations of data sets such that models trained on coresets, are provably competitive with models trained on the full data set. The lightweight coreset approach [33] relaxes the traditional core-set approach by introducing an additive term (to scale with variance) in addition to the multiplicative term in traditional coresets. This coreset is then used to perform k-means clustering.

5.2.9 ContrastiveActiveLearning (time-taking) Contrastive Active Learning [34] selects instances with the maximum mean Kullback-Leibler (KL) divergence between the predicted class distributions ("probabilities") of an instance and each of its m nearest neighbours:

$$\arg\max_{x_i} \left[\frac{1}{m} \sum_{j=1}^{m} KL(P(y_j|x_j^{knn}) \| P(y_i|x_i) \right]$$

; where the instances x_j^{knn} are the m nearest neighbours of instance x_i

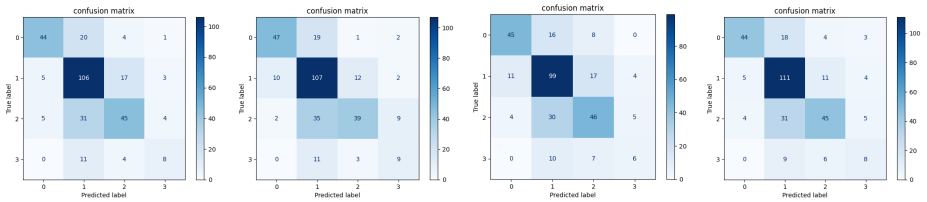
As pointed out in [35], transformers are not suitable for most of the query strategies, DL ensembles are too expensive and prediction entropy is too overconfident [36] Exceptions are flat architectures eg FastText [37]. Before transformers, QS based on Expected Gradient length was norm, but they scaled with a vast number of transformer parameters and needed computations per instance wise (and not per batch wise).

Experimental Setup We run Active Learning strategies with on preprocess_5 (best performing for distilbert). We use GPU A100 on Google Colab (for faster execution). We set the batch to 20 samples for each query and run till 35 queries.

6 Results

6.1 Normal Text Classification Methods

Before using applying, it would be helpful to get an idea of the regular text classification methods that are used. This analysis will also enable us to select a base model for Active Learning strategies. We use V100 GPU on Google Colab for our experiments.

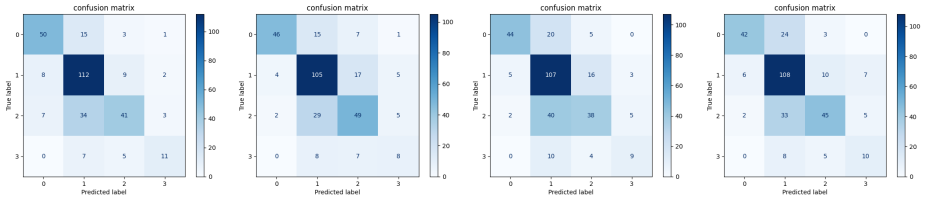


(1) Acc : 65.91%, BA : 57.75 ± 1.11%, AUC-ROC : 0.8126 ± 0.0047

(2) Acc : 65.58%, BA : 58.36 ± 0.76%, AUC-ROC : 0.8214 ± 0.0097

(3) Acc : 63.64%, BA : 55.31 ± 0.14%, AUC-ROC : 0.8154 ± 0.0021

(4) Acc : 67.53%, BA : 58.74 ± 0.70%, AUC-ROC : 0.8188 ± 0.0022



(5) Acc : 69.48%, BA : 63.51 ± 0.01%, AUC-ROC : 0.8147 ± 0.0001

(6) Acc : 67.53%, BA : 59.81 ± 0.01%, AUC-ROC : 0.8113 ± 0.0001

(7) Acc : 64.29%, BA : 57.32 ± 0.01%, AUC-ROC : 0.8009 ± 0.0001

(8) Acc : 66.56%, BA : 60.00 ± 0.15%, ROC-AUC : 0.8287 ± 0.0003

Fig. 7: Output Scores of DistilBERT along with Accuracy (from the last run), Balanced Accuracy (averaged over 5 runs) and ROC-AUC Scores (averaged over 5 runs)

Distilbert (distilbert-base-german-cased) Results: We conclude the following from the results shown above:

1. The untagged preprocessing (plain-text) performs better than tagged preprocessing (possibly because distilBERT was neither trained nor fine-tuned to german-base-cased using similar tags)
2. The difference among various preprocessing is not much. For tagged preprocessing, the difference in BA is approximately 2.26% and for ungagged, it is around 2.92% points.
3. Different runs give slightly different results. Owing to different initialisations and dropouts.
4. Though it is difficult to generalise the results from different runs since the difference in BA is very less, we can still deduce the following about order-based preprocessing :

– The Empirically found order (preprocessing 1,4) and low-to-high order (preprocessing 2,5) result in a higher BA.
– The high-to-low order and order of importance generated by Extra-trees classifier do not work well distilBERT.

Table 1: HAHNN with GRU

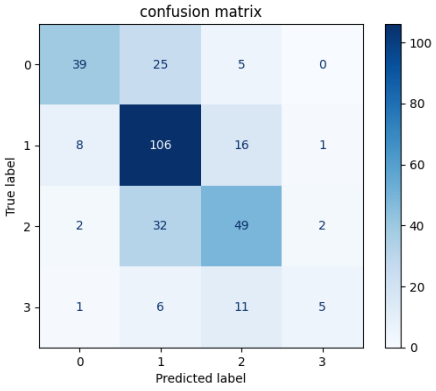
PP	BA	ROC-AUC
1	$50.66 \pm 1.85\%$	0.7594 ± 0.0070
2	$48.47 \pm 1.37\%$	0.7686 ± 0.0132
3	$50.60 \pm 1.17\%$	0.7809 ± 0.0104
4	$50.70 \pm 1.18\%$	0.7724 ± 0.0094
5	$47.73 \pm 1.28\%$	0.7448 ± 0.0057
6	$47.87 \pm 2.53\%$	0.7633 ± 0.0074
7	$53.79 \pm 1.33\%$	0.7858 ± 0.0029
8	$47.92 \pm 3.83\%$	0.7660 ± 0.0165

Table 2: HAHNN with LSTM

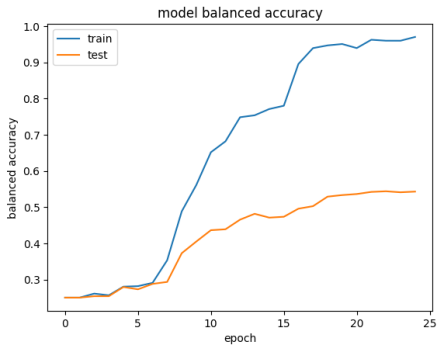
PP	BA	ROC-AUC
1	45.78 ± 3.66	0.7443 ± 0.0195
2	49.67 ± 1.19	0.7681 ± 0.0053
3	50.11 ± 3.12	0.7728 ± 0.0209
4	55.46 ± 3.23	0.8060 ± 0.0128
5	49.16 ± 2.51	0.7613 ± 0.0093
6	54.89 ± 0.67	0.8011 ± 0.0118
7	50.92 ± 1.95	0.7820 ± 0.0141
8	53.31 ± 2.35	0.8012 ± 0.0075

HAHNN Results: We present the results in Table 1 and the confusion matrix in Figure 9 (a). We notice the following trends here:

1. Scores for LSTM are in general slightly higher than those of GRU.
2. Preprocessing with tags clearly outperforms the preprocessing without tags (at least by 2% difference) for GRU. However, this is reversed for LSTMs.
3. The preprocessing 7 (tagged and ordered by decreasing importance) clearly outperforms other preprocessing for GRU. For, LSTM it achieves good enough performance.
4. Overfitting to train data (Figure 9 (b)).
5. HAHNN first learns to assign the majority class, then it gradually learns other classes. It would be interesting to study why this happens and maybe use it to handle other methods which were affected by imbalanced data.



(1) Confusion Matrix on preprocess_7 for HAHNN (GRU) on test set.



(2) BA vs epoch curve for a run on preprocess_7 for test set

Fig. 8: Results of HAHNN for preprocess_7 and the plot of Balanced Accuracy showing overfitting

XLNet After running the model on the A100 GPU on Colab Pro, we found that XLNet is not actually suitable for the dataset with a moderate imbalance. As seen from the confusion matrix obtained by running preprocess_1, the model essentially learned the majority class, and consequently, the balanced accuracy is close to 0.25 and the ROC-AUC score is 0.4963 (no better than a random classifier). We found that, unlike HAHNN, XLNet learns to assign the majority class to all classes and this does not change with epochs. This leads us to abandon the model. The unsuitability of XLNet to imbalanced data is well documented in the literature.

Electra We thought that the model might be better at learning contextual relations given the same computational resources as Distilbert. However using V100 GPU and full-length text, we found that the model ends up learning only the majority class (similar to XLNet).

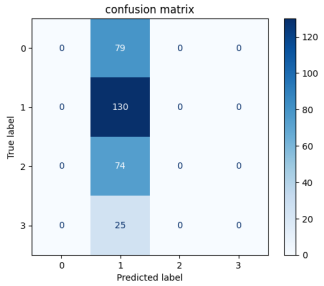


Fig. 9: Confusion Matrix on XLNet or Electra

6.2 Active Learning Methods

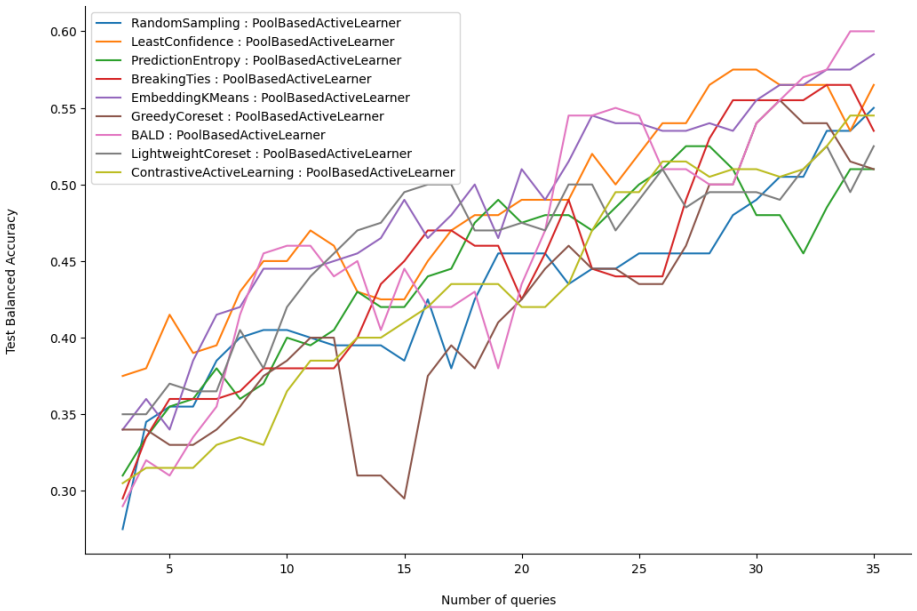


Fig. 10: Balanced Accuracy plots obtained for preprocessing_5 (best output for distilbert)

We observe the following about the various discussed methods:

1. the trend for almost all curves is steeper at the start and converging towards the end. (the 4-median-smoothening makes the curve a bit more linear though)
2. Random Sampling starts catching up and is able to outperform some other query strategies later on
3. Prediction Entropy is expected to perform suboptimally since it's based on uncertainty estimates (logits) and neural networks are notorious for their uncertainty estimates.
4. BALD, Least Confidence and EmbeddingKMeans are top-performing query strategies for the most part
5. Coreset-based methods (GreedyCoreset and LightweightCoreset) provide decent enough performance (with Lightweight corset giving better performance due to an additional additive error term in its objective).
6. We can quantify the minimum size of the labelled pool based on the top-3 query strategies: BALD, Least Confidence and Embedding-K-Means

Balanced Accuracy	**Num. of Queries**	**% of training data**
45%	10	16.58%
50%	20	33.17%
55%	25	41.46%
60%	35	58.04%

Table 3: Bounding the minimum size of the labelled pool for top 3

7 Conclusion

From the above experiments, we can conclude that we can get 63% Balanced Accuracy using Normal Classification methods (BERT, DistilBERT, HAHNN). The most optimal pre-precessing turns out to be low-to-high frequency untagged concatenation of textual-features.Using Active Learning, we can get close to 45% Balanced Accuracy by labelling just 17% data points and 60% Balanced Accuracy (close to full dataset) by labelling 60% data points.

8 Future Works

We propose the following future directions :

On objectives: Using similar techniques, we might predict the estimated duration.

On preprocessing and Data-handling

1. We missed out on 4 forms : SEPERATE_TEXT, DND_ORDER, DND_IN_TEXT and OPEN. Among all 4, the OPEN type needs special emphasis to be included in the processing because it consists of a lot of examples.
2. Explore using NLP to predict difficulty of mathematical questions (by form).

On approaches: The following aspects could be further explored as an extension of approaches presented in the above work:

1. Imbalance-handling methods to render models like XLnet and Electra useful.
2. Ordinal regression to reduce the number of queries (information in unlabelled instances based on metrics such as similarity).
3. Using full-text models like BigBird, or averaging in instead of chunking
4. Incorporating Ordinal Regression in Model for Active Learner
5. Design research based on Argilla interface
6. Dynamic AL strategies eg DUAL, GraDUAL and frameworks like AcTune

9 References

References

1. Benedetto et al., "On the application of Transformers for estimating the difficulty of Multiple-Choice Questions from text", https://aclanthology.org/2021.bea-1.16/
2. "Papers with Code: Datasets and Benchmarks", https://paperswithcode.com
3. "A Survey on Recent Approaches to Question Difficulty Estimation from Text", https://dl.acm.org/doi/10.1145/3556538
4. Ronald K. Hambleton and Russell W. Jones, "Comparison of classical test theory and item response theory", 1993.
5. Ronald K. Hambleton, Hariharan Swaminathan, and H. Jane Rogers, "Fundamentals of Item Response Theory", Sage, 1991.
6. Georg Rasch, "Probabilistic Models for Some Intelligence and Attainment Tests", Danish Institute for Educational Research, 1960.
7. Settles, 2009.
8. "Imbalance handling methods (blog)", https://neptune.ai/blog/how-to-deal-with-imbalanced-classification-and-regression-data
9. Niekler and Schroder, 2020.
10. Zhang et al., 2017.
11. Uncertainty Sampling - Lewis and Gale, 1994.
12. Entropy-based output Uncertainty - Shannon, 1948.
13. Least Confidence - Culotta and McCallum, 2005.
14. Margin Sampling - Scheffer et al., 2001; Schein and Ungar, 2007.
15. Schröder et al., 2022.
16. Query-by-committee (QBC; Seung et al., 1992).
17. Settles et al., 2007.
18. Expected Loss Reduction - Roy and McCallum, 2001.
19. BERT - Devlin et al., 2019.
20. Transformers - Vaswani et. al, 2017.
21. DistilBERT - Sanh et al., 2019.
22. Hinton et al, 2015.
23. HAHNN - paperswithcode.com
24. XLNet - paperwithcode.com
25. Electra - paperswithcode.com
26. Small-text : https://github.com/webis-de/small-text
27. Least Confidence - (LC; Culotta and McCallum, 2005).
28. Prediction Entropy - PE; Roy and McCallum, 2001; Schohn and Cohn, 2000.
29. BT; Scheffer et al., 2001; Luo et al., 2005.

30. Niekler and Schroder, 2020.
31. Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. "Cold-start Active Learning through Self-supervised Language Modeling", In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) Association for Computational Linguistics, pages 7935–7948, 2020.
32. Ozan Sener and Silvio Savarese. "Active Learning for Convolutional Neural Networks: A Core-Set Approach", In International Conference on Learning Representations 2018 (ICLR 2018), 2017.
33. Olivier Bachem, Mario Lucic, and Andreas Krause. "Scalable K-Means Clustering via Lightweight Coresets", In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18). Association for Computing Machinery, New York, NY, USA, 1119–1127, 2018.
34. CA; Margatina et al., 2021.
35. "Revisiting Uncertainty-based Query Strategies for Active Learning with Transformers", 2022, [Paper] [Source code].
36. Guo et al., 2017, Lakshminarayan et al. 2017.
37. Prabhu et al., 2019.
38. "Imbalance Handling approaches", https://www.linkedin.com/pulse/what-imbalanced-dataset-its-impacts-machine-learning-models-cheruku/
39. "Shanon Entropy", https://stats.stackexchange.com/questions/239973/a-general-measure-of-data-set-imbalance
40. "SMOTE", https://arxiv.org/pdf/1106.1813.pdf
41. "ADASYN", https://www.youtube.com/watch?v=mKG7lnZNAOk
42. "FastText", https://fasttext.cc/docs/en/crawl-vectors.html