The 35th Conference on Computational Linguistics and Speech Processing (ROCLING 2023)
Taipei City, Taiwan, October 20-21, 2023. The Association for Computational Linguistics and Chinese Language Processing

Category Mapping for Zero-shot Text Classification

Qiu-Xia Zhang*
Department of Computer Science and Information Engineering
National Taiwan University
r10922164@ntu.edu.tw
Te-Yu Chi*
Department of Computer Science and Information Engineering
National Taiwan University
d09922009@ntu.edu.tw
Te-Lun Yang*
Department of Computer Science and Information Engineering
National Taiwan University
d12944007@ntu.edu.tw
Yu-Meng Tang*
Department of Computer Science and Information Engineering
Tongji University
tonmoregulus@gmail.com
Ta-Lin Chen*
Department of Electrical and Computer Engineering
University of Texas at Austin
talin@utexas.edu
Jyh-Shing Roger Jang
Department of Computer Science and Information Engineering
National Taiwan University
jang@mirlab.org

Abstract

The existing method of using large pre-trained models with prompts for zero-shot text classification possesses powerful representation ability and scalability. However, its commercial availability is relatively limited. The approach of employing class labels and existing datasets to fine-tune smaller models for zero-shot classification is comparatively straightforward, yet it might lead to weaker model generalization ability. This paper introduces three methods to enhance the accuracy and generalization capability of pre-trained models in zero-shot text classification tasks: 1) utilizing pre-trained language models and structuring inputs into a standardized multiple-choice format; 2) creating a text classification training dataset using Wikipedia text data and refining the pre-trained model through fine-tuning; and 3) suggesting a zero-shot category mapping technique based on GloVe text similarity, wherein Wikipedia categories replace textual categories. Remarkably, without employing labeled samples for fine-tuning, the proposed method achieves results comparable to the best models fine-tuned with labeled samples.

Natural Language Processing, Pre-trained Language Models, Zero-shot Text Classification, Classification, GloVe

1 Introduction

Text classification is a pivotal task within the realm of natural language processing, with extensive applications in areas such as spam filtering, information retrieval, personalized recommendations, sentiment analysis, and pub-

*These authors contributed equally to this work.

lic opinion monitoring. Presently, pre-trained models, after fine-tuning on labeled data, have achieved substantial accuracy improvements on these labeled datasets. However, there are inherent limitations when relying solely on supervised methods in practical applications. One primary concern is the necessity to construct a new dataset for each novel new task, involving significant data collection and manual annotation efforts. This process consequently escalates both time and labor costs. Particularly containing multiple tasks of text classification, incomplete data collection may lead to issues associated with data sparsity.

The zero-shot classification model stands out due to its cross-domain universality, no need for manual labeling for new tasks, thus considerably saving time and lobar costs. Two primary approaches are currently prevalent for zero-shot classification. One employs the prompt (Brown et al., 2020) method, leveraging the contextual attention mechanism of large pre-trained language models, using prompts to guide the model to generate the desired responses. However, the leverage of these large models often depends on commercial APIs like GPT and ChatGPT, restricting independent commercial usage.

An alternative strategy capitalizes on existing open-domain datasets or uses the unlabeled data or labels from the target dataset to create training data, fine-tuning smaller pre-trained language models. Nevertheless, due to the lesser parameters learned by these smaller models, there exists a challenge of weaker generalization capabilities.

This study primarily aims to address the weaker generation capacity of small pre-trained language models, enhancing their perceptions for classification tasks and further facilitating the knowledge transfer from pre-trained language models to target datasets. The contribution of this paper is the proposition of a category mapping method based on GloVe text similarity, integrated with the UniMC (Yang et al., 2022) model fine-tuned on wiki data. This approach has yielded results on par with the state-of-the-art methods in zero-shot text classification tasks.

2 Related Work

GloVe (Global Vectors for Word Representation) aims to preserve both syntactic and semantic word relationships while enhancing the effectiveness of word vector clustering. GloVe synergistically incorporates the advantages of both Latent Semantic Analysis (LSA) (Dumais et al., 2004) and Word2Vec (Mikolov et al., 2013). It employs training on co-occurrence matrices to proficiently capture semantics through global statistical insights. Within this research, the calculation of text similarity hinges on the GloVe model.

Prompt models like GPT-3 (Brown et al., 2020) utilize predefined prompts to guide downstream tasks by constraining the model's output. These prompts, expressed naturally, enable the model to complete tasks effectively. Both InstructGPT (Ouyang et al., 2022) and ChatGPT use instruction-tuning techniques, leveraging prompts to influence text generation and fill in gaps. However, designing prompts requires specialized knowledge due to the lack of universal templates. Generative upstream models might introduce irrelevant content. Additionally, large-scale language models' speed relies on efficient API calls, limiting their practical use in commercial applications.

TE-Wiki (Ding et al., 2022) (Textual Entailment formulation with Wikipedia fine-tuning) utilizes open-source Wikipedia text to construct training data. It employs Wikipedia text as premises and Wikipedia categories as hypotheses, formatted according to "[Text] Entails [Label i]" for i [n], to perform binary classification on whether a certain text entails a particular category.

Zero-Shot Text Classification with Self-Training (Gera et al., 2022) employs a methodology based on Natural Language Inference (NLI). Unlike TE-Wiki, the article utilizes training data consisting of unlabelled data to be predicted for iterative training.

UniMC (Yang et al., 2022) employs a self-attentive encoder structure that transforms label-based natural language understanding (NLU) tasks into a unified multiple-choice format. Labels are treated as options, and a token [O-MASK] is introduced before each option to predict the probability of selecting that option. The model is trained using 14

NLU task datasets and fine-tuned on the pre-trained ALBERT model. During fine-tuning, the model computes softmax over the 'yes' logits for each [O-MASK] output, determining the probability of each option. The option with the highest probability is used for prediction. The cross-entropy loss is calculated between the predicted answer and the standard answer. UniMC takes both the content and class labels of the text into consideration, yielding more accurate text representations. This enables better expression of the relationship between text and categories, as well as the relationships between different categories. Furthermore, the input is structured as a multiple-choice format, and the concept of prompts is incorporated during text processing, enhancing the model's awareness of classification tasks and improving its accuracy in handling such tasks.

3 Datasets Introduction

The experiment involves four types of text classification datasets, as shown in Figure 5. Among them, Yahoo Answers Topic, AG News, and DBPedia are topic classification datasets, while imdb is an emotion classification dataset. The labels in all these datasets are evenly distributed. As zero-shot datasets, we use the test sets from the above datasets to measure the performance of the model. The evaluation metric is based on the accuracy on these test sets.

4 Methodology

The experimental methodology primarily involves three stages: preprocessing of the training data, model fine-tuning, and post-processing.During the preprocessing of training data, this experiment utilizes open domain text data from Wikipedia webpage, and structure it into the input format required by the UniMC model. This is then used to fine-tune the UniMC model. Lastly, during the inference stage, we employ a category mapping method based on the GloVe model to replace the target label with the wiki label used during the fine-tuning process, which in turn enhances the predictive accuracy of the model.

Algorithm 1: Training Data Collection

input : Top-level category set S,Wikipedia subcategory graph $\mathcal{G}$, Wikipedia articles X, max search depth $r = 2$;
output: M

Initialize $d(x, c) = \infty$ for any article $x \in X$ and $c \in S$ · $M = \{\}$;
for c in S **do**
  T = DFS($c, \mathcal{G}, r$);
  for t in T.nodes **do**
    for x in t.articles **do**
      $d(x, c) = \min\{d(x, c), 1 + \text{depth}(t)\}$
for x in X **do**
  if $\min_{c \in S} d(x, c) < \infty$ **then**
    $P = \text{argmin}_{c \in S}\, d(x, c)$;
    for c in P **do**
      if c. len() == 1 **then**
        Add $(x, c, 1)$ to M;
        for i in $[1, n-1]$ **do**
          Sample c' from $c' \in S - P, c' \neq P \cup \{c'1, c'2, \ldots, c'n-2\}$;
          Add $(x, c', 0)$ to M;

Figure 1: Algorithm for wiki-collect

The following will provide an introduction to these three steps.

4.1 Acquisition of Open-Domain Training Data

Firstly, let's discuss the preprocessing of the training data. In this experiment, the method of TE-Wiki is referenced and modified to construct a category tree for Wikipedia. This category tree is then used to build the training data. From the 700 top-level categories on Wikipedia, we removed 26 categories that began with "List of." We then used the remaining 674 categories as the root nodes of the categories tree. Using depth-first search with a set depth of 2, we identified all subcategories of these root nodes. These subcategories were used as nodes to create the categories tree. Once the categories tree was constructed, we located all articles directly under the root nodes, meaning articles belonging to only one category.

We chose these articles as training texts. Unlike TE-Wiki, which only constructs two sample differences, for every piece of data in our study, we constructed classification samples with n categories. We selected the category to which these articles belonged as the positive label and then randomly chose n-1 labels from the remaining 673 labels as negative labels. The algorithm is described in 1.

Finally, we organized the labels into tuples containing text and other elements. During the fine-tuning phase, these tuples were structured into the input format required by various models.

4.2 Model Input Formatting

For generative models using prompts, such as GPT-3.5, the training data is formatted as:

$$\text{“[class}_i\text{],” for class}_i \in \text{class\_list [prompt] [text]} \quad (1)$$

For the TE-Wiki model, each instance is formatted as:

$$\text{“[CLS][text][SEP][class }_i\text{] [SEP]”for class}_i \in \text{class\_list} \quad (2)$$

For models using the Self-training method, each instance is formatted as:

$$\text{“[CLS][text][SEP][prompt + class }_i\text{] [SEP]” for class }_i \in \text{class\_list} \quad (3)$$

For models utilizing a unified multiple-choice format, each instance is structured as:

$$\text{[CLS] “([O} - MASK_i\text{] [class}_i\text{] for i} \in \text{n)” [SEP][prompt] [SEP] [Text] [SEP]} \quad (4)$$

For models that require the use of prompt words, apart from Self-training where we retained the prompt used in the original paper “This example is” , other models in this study use the unified prompt: “Which category does the following text belong to?” .

4.3 Category Mapping

Before performing model inference, we select the Wikipedia category most similar to each target category to build a synonym list. During the model interference process, we use words from the synonym list to replace the target category for predictions. Specifically, this process includes the following steps:

0. Preprocessing for Category Mapping: Assume the category string to be inputted is s containing words w_1, w_2,...,w_n.

1. Using the GloVe model, compute the word vectors for both the target category and Wikipedia category.

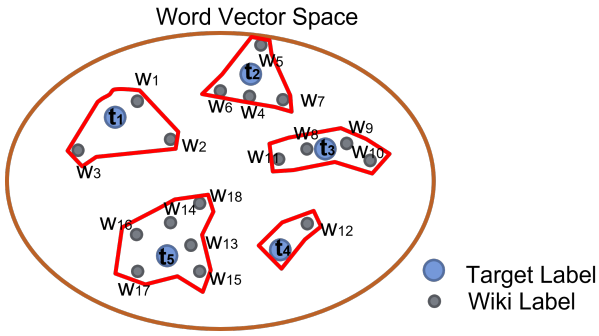


Figure 2: Schematic diagram of classifying Wikipedia categories in word vector space based on target categories

2. Based on the cosine similarity between the word vectors of the Wikipedia category and the target category, the Wikipedia categories are mapped to the synonym list of the target category. For each Wikipedia category, we calculate its cosine similarity with the word vector of every target category, resulting in a similarity matrix between the Wikipedia categories and the target categories. $S \in \mathbb{R}^{W \times Z}$, S=

$$\begin{bmatrix} s(w_1, z_1) & s(w_1, z_2) & \cdots & s(w_1, z_{n_z}) \\ s(w_2, z_1) & s(w_2, z_2) & \cdots & s(w_2, z_{n_z}) \\ \vdots & \vdots & \ddots & \vdots \\ s(w_i, z_1) & s(w_i, z_2) & \cdots & s(w_i, z_j) \end{bmatrix}$$

Where $S_{i,j}$ represents the cosine similarity between the Wikipedia category i and the target category j. For each Wikipedia category, we select the most similar target category k, and add this Wikipedia category to the synonym list M_k of its most similar target category. The mapping function can be represented as:

$$f(i) = k = \arg\max_j S_{i,j} \quad (5)$$

In the word vector space, the Wikipedia categories are classified according to the target categories as shown in Figure 2. In this example figure, w_1, w_2,...,w_8 are the Wikipedia categories to be classified. For w_1, w_2,w_3, the target category most similar to them is c_1, so they are added to the synonym list of c_1.

As shown in Figure 3, after mapping each Wikipedia category to the list of synonym candidates for the target category, the candidates in each target category’ s synonym list are sorted based on their similarity to the target

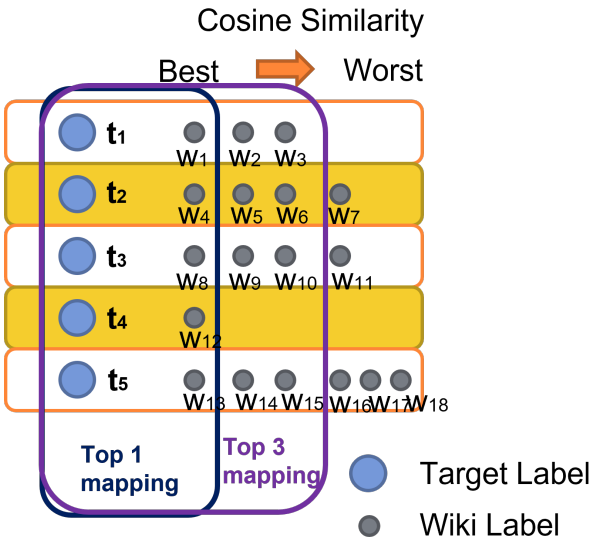


Figure 3: Schematic diagram of mapping Wikipedia categories to target category candidates

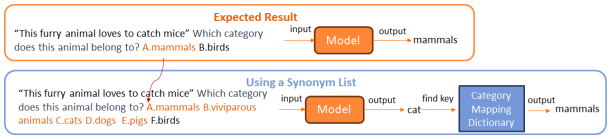


Figure 4: Diagram for the Use of Synonyms

category. Then, according to the practical requirements, the top k candidate words are selected, which are the top k Wikipedia categories most similar to the target category, to be the options in the final synonym list.

3. During the model inference process, synonyms are used to replace the target categories for prediction. After the final list of synonyms is obtained, during inference predictions, these Wikipedia categories are used to replace the target categories as input, allowing the fine-tuned model to classify these categories. As shown in Figure 4, suppose the *text* to be predicted is "This furry animal loves to catch mice", the *class_list* is ["mammals", "birds"], and the *ground_truth* is "mammals". During the inference process, the synonyms list, which includes ["homeothermic vertebrates", "live-birth animals", "cats", "dogs", "pigs", "rabbits"], is used to replace "mammals"as options, and is input into the model for inference based on these synonyms. To avoid poor-quality synonyms and eliminate the interference on the model's judgement caused by non-similar words in the synonym list, this study also added a filtering mechanism to the synonyms during the experiment. The implementation method consists of two aspects: 1. Setting a threshold 2. Confirmation of the target category word. Firstly, to exclude Wikipedia categories with low similarity to the target category, a similarity threshold of 0.8 is set. If the cosine similarity between the Wikipedia category and the target category in the synonym candidates is less than 0.8, that candidate is deleted. Secondly, to ensure finding synonyms with high similarity in Wikipedia categories, a mechanism for confirming the target category word is introduced. This confirmation mechanism determines whether the lowercase of the target category word is a substring of the lowercase of a synonym. If so, there's no need to add it; if not, the word should be included.

4. In post-processing, the model's output is mapped to the target category based on the category mapping dictionary. After the model completes the inference, the synonyms dictionary is used. The model's output is used as the value to search for its corresponding key, and the key is then output as the final result. As shown in Figure 4, after using the synonyms list, the model determines that the animal is a "cat"based on the sentence context. Then, the value "cat"can be matched with its key "mammals" in the synonyms dictionary. Therefore, "mammals" is output as the final answer.

5 Experiments

For the aforementioned methods, this study set up four sets of experiments:

1. Experiment Group 1 was set up to explore the performance of various zero-shot classification models and to confirm the feasibility of the UniMC model.

2. Experiment Group 2 aimed to compare the results before and after fine-tuning the UniMC with training data constructed from

Wikipedia webpage text. This was to determine the viability of the method. In addition to the result comparison experiment, another test was set up to evaluate the effect of the number of training data categories (n) on the fine-tuning result and to select the optimal category count (n).

3. Experiment Group 3 investigated the effect of category mapping. Experiment 3.1 was designed to explore the effects of different synonym quantities (k). Experiment 3.2 aimed to study the effects of the screening mechanism.

4. Experiment Group 4 conducted an ablation study to understand the relationship between the Wikipedia fine-tuning and category mapping methods, as well as their impact on the model's performance.

Using the best model determined through the above methods, in Experiment 5.1 and Experiment 5.2, this study compared its performance with the pre-tuned original UniMC model and the current best-performing Self-training model.

5.1 Experimental Setup

In this study, experiments were conducted using the PyTorch development container (Model: cm.xsuper) provided by the National Supercomputing Center's Taiwan Computing Cloud (TWCC). The experimental environment settings are shown in Table 7.

5.2 Zero-shot Text Classification Model Performance Comparison

This experiment set up multiple models with different architectures to laterally assess the performance of UniMC and other models in zero-shot classification. The experiment was divided into four control groups: the UniMC model, the GPT-3.5 model with prompts, TE-Wiki, and the Self-training model. The parameters for each model are detailed in Table 7.

Among the aforementioned models, except for GPT-3.5, which was inferred by invoking its API on an item-by-item basis, the other models used a batch size of 16 during inference.. It's worth noting that for answers generated by GPT-3.5, if the output does not contain a category, the cosine similarity between the generated content and the target category is calculated in the GloVe word vector space. The most similar category is then selected as the output.

The final results of the experiment are presented in Figure 8. The shown values represent accuracy percentages. As the DBpedia dataset is large, this study did not use GPT-3.5 to infer on it. Bold parts in the figure denote the highest scores on a particular dataset, while pink parts indicate where the UniMC model achieved the best scores. The figure clearly shows that the Self-training model has the best average performance, achieving the highest scores on both the AG News and DBpedia datasets. Although the UniMC model performed poorly on the DBpedia dataset, it outperformed other models on the Yahoo Answer and IMDB datasets. This suggests that the key to improving the UniMC model's efficiency lies in increasing its accuracy on datasets like DBpedia.

5.3 Comparison of Model Performance Before and After Fine-tuning using Wikipedia Data

To investigate the effectiveness of fine-tuning the UniMC model using training data constructed from Wikipedia web page text, Experiment 2.1 was set up to compare the results before and after this fine-tuning. During the training process, the experimental parameters were set as follows: batch size of 4, a learning rate of $2e-5$, early stopping with a patience value of 5, saving a checkpoint every 500 steps, and the optimizer being AdamW. For training data processing, 9,749 entries from the training data were chosen as the validation set, accounting for 0.01%; the remaining 965,174 entries served as the training set, making up 0.99%. The number of categories n was set to 5.

The results, as shown in the bar chart 9, reveal that the accuracy of the fine-tuned model slightly declined on the IMDB dataset. However, there was a marked improvement on the other three datasets. Specifically, the accuracy on the DBpedia dataset jumped from 12.93% to 68.02%, which was the most no-

ticeable enhancement. After fine-tuning the UniMC with Wikipedia data, the average accuracy improved by 14.07%, validating the efficacy of this approach.

5.4 Exploration of Optimal Number of Training Data Categories

To ascertain the ideal number of categories in the training data and whether this number impacts the model's performance, control groups were set up for this experiment. These were UniMC-5 classes, UniMC-10 classes, UniMC-20 classes, UniMC-30 classes, UniMC-40 classes, and UniMC-50 classes. These represent training data with a total category count n of 5, 10, 20, 30, 40, and 50, respectively. All these training datasets utilized single-category text data, meaning only one positive sample, with negative samples labeled with 4, 9, 19, 29, 39, and 49 labels, respectively.

The results, as shown in the bar chart 10, did not indicate a clear correlation between category count and model performance. However, based on these findings, the optimal model chosen for further experimentation was the one fine-tuned with 40 categories. Subsequent experiments will be based on this model.

5.5 Synonym List Effect Exploration

To investigate the impact of using synonyms and the number of synonyms on model prediction results, this experiment selected the top k synonyms, with k being 1, 3, 5, 7, and 9, respectively. During the inference process of the model, the target category is replaced with these synonyms to observe the effects of various synonym list lengths. The GloVe.6B model was used to generate the target category and Wikipedia category. This model is pretrained on six million tokens and includes corpora from Gigaword5 and Wikipedia2014. It has an output vector dimension of 300.

The results are shown in the bar chart 11. The chart clearly shows that, except for the AG News dataset, the model prediction results significantly decreased after using the synonym list. Further analysis is needed to determine the cause.

From the Yahoo Answers dataset, four categories were randomly selected. Their synonym lists and similarities between the synonyms and target category when taking the top 9 synonyms were extracted, as exemplified in 6. Analysis revealed that among the top 9 synonyms for each target, there could be Wikipedia categories with very low similarity to the target category, some even below 0.6. Considering that simply using the top k synonyms might interfere with model predictions, a synonym screening mechanism was introduced, leading to Experiment 3.2.

5.6 Screening Mechanism Effect Experiment

The experiment purpose is to introduce a screening mechanism to eliminate the negative impact of dissimilar synonyms on model performance and to explore the effectiveness of this mechanism. We utilized the synonym screening mechanism mentioned in method 4.3, setting a threshold of 0.8, and confirmed the target category. The number of synonyms taken was 5, 7, and 9, respectively. Results before and after using the screening mechanism were compared, as displayed in 12.

After incorporating the screening mechanism, the average synonym length for each category is shown in table 1. The synonym list's average length is reduced to 1-3 Wikipedia categories corresponding to each target word, which is less than taking the top k. The average performance after implementing the screening mechanism improved by 15.12%. Except for $k = 5$ and $k = 7$ on AG News, where the performance slightly decreased, the results with the screening mechanism surpassed those without it. Given that the best result was obtained with $k = 5$ using the screening mechanism, we designated this model as our final model, naming it UniMC-Wiki.

5.7 Wikipedia Fine-tuning and Category Mapping Ablation Study

This experiment employed ablation studies to analyze the impact of Wikipedia fine-tuning and category mapping on model performance, as well as the interaction between these two methods. In this experiment, the following four control groups were set up: "UniMC-ori", "UniMC-ori, label mapping", "UniMC-40

classes", and "UniMC-40 classes, label mapping".

"UniMC-ori" represents the model without Wikipedia data fine-tuning, with inference on target categories.

"UniMC-ori, label mapping" represents the model without Wikipedia data fine-tuning, with inference on Wikipedia categories within the alternate word list after category mapping.

"UniMC-40 classes" represents the model fine-tuned with Wikipedia data, with inference on target categories.

"UniMC-40 classes, label mapping" denotes the model fine-tuned with Wikipedia data, with inference on Wikipedia categories within the alternate word list after category mapping.

The model that was fine-tuned with Wikipedia data was adjusted with Wikipedia categories where the value of n was 40. In the category mapping method, the top $k = 5$ most similar words were selected and incorporated into a filtering word mechanism.

The results of the experiment are presented in a bar chart in Figure 13. Comparing the first experimental group with the third and fourth groups, it can be seen that even without using category mapping, fine-tuning with Wikipedia data can improve the prediction accuracy of the model. This improvement in model performance is independent and does not rely on other factors. Comparing the first experimental group with the second and fourth groups respectively, it is evident that category mapping can only have a positive effect on the model if it has been fine-tuned using Wikipedia data; otherwise, it can lead to a decrease in model performance. Comparing the average accuracy of all experimental groups, it can be found that performing both fine-tuning and category mapping achieves the best results.

5.8 Performance Comparison Experiment Before and After using the Research Method

Compared the final model, UniMC-Wiki, with the pre-fine-tuned model, UniMC-ori. The bar chart of the results is shown in Figure 14. It can be observed that after employing the research method, performance on all datasets except for the IMDB dataset has significantly increased. Specifically, on the DBpedia dataset, the accuracy using the research method increased by 80.76% compared to not using it. In terms of overall performance, compared to the original model, the UniMC-Wiki's average accuracy improved by 22.14%.

To investigate why the research method performed relatively poorly on sentiment classification tasks, we extracted the alternate word list from the final model on the IMDB dataset. We found that the labels in this alternate word list were "positive" and "negative", which are consistent with the original target categories. This indicates that in the Wikipedia categories, there aren't words with high similarity to the sentiment category labels "positive" and "negative". As Wikipedia is topic-oriented, it has certain limitations when it comes to sentiment-related tasks. Therefore, the method was not effective on the IMDB dataset.

5.9 Performance Comparison Experiment between UniMC-Wiki and the Best Model

In this experiment, a performance comparison was conducted between UniMC-Wiki and the current best model, Self-training. The experiment was set up with a batch size of 16 for model inference. The results of the experiment are shown in Figure 15.

The final experimental results showed that the UniMC-Wiki model performed better than the Self-training model on the Yahoo Answers and AG News datasets. Conversely, the Self-training model performed better on the other two datasets. The average accuracy of the UniMC-Wiki model was slightly higher than Self-training by 0.61%, achieving results comparable to the best model.

6 Conclusion and Future Works

This paper employs three methods to address the problems of data scarcity and domain dependence in zero-shot classification tasks: training the model with the UniMC structure, fine-tuning the UniMC structured model using Wikipedia to build classification task training data, and utilizing category mapping. Five sets of experiments were designed to validate the feasibility of these methods. The final experiments demonstrated that using the methods proposed in this paper achieved a 22.14% improvement compared to before. Moreover, the methods in this paper achieved results comparable to the current best self-training model on average. We found that it significantly enhanced the performance of topic classification tasks. However, its effect on sentiment classification tasks was not evident.

Based on the above conclusions, we believe that future work should focus on further exploring how to find more suitable knowledge sources for sentiment classification, investigating how to introduce their knowledge into the model more effectively. Additionally, there's room to further improve the UniMC model structure and design a mechanism that can automatically select prompts during the training process, thereby enhancing the model's performance and accuracy.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Hantian Ding, Jinrui Yang, Yuqian Deng, Hongming Zhang, and Dan Roth. 2022. Towards open-domain topic classification. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations*, pages 90–98.

Susan T Dumais et al. 2004. Latent semantic analysis. *Annu. Rev. Inf. Sci. Technol.*, 38(1):188–230.

Ariel Gera, Alon Halfon, Eyal Shnarch, Yotam Perlitz, Liat Ein-Dor, and Noam Slonim. 2022. Zero-shot text classification with self-training. *arXiv preprint arXiv:2210.17541*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Ping Yang, Junjie Wang, Ruyi Gan, Xinyu Zhu, Lin Zhang, Ziwei Wu, Xinyu Gao, Jiaxing Zhang, and Tetsuya Sakai. 2022. Zero-shot learners for natural language understanding via a unified multiple choice perspective. *arXiv preprint arXiv:2210.08590*.

7 Appendix

Item	Parameters
CPU	Intel(R) Xeon(R) Gold 6154 (8 cores)
GPU	Nvidia Tesla V100 * 2
RAM	128 GB
OS	Ubuntu 20.04 LTS

type=tabletableExperimental Environment

	AG News	Yahoo Answer	DBpedia	IMDB
Top 5	2	2.2	1.071	1
Top 7	2.5	2.6	1.071	1
Top 9	2.5	2.7	1.071	1

Table 1: Average synonym list length per category after adding filtering mechanism

Dataset Name	Dataset Attribute	Number of Categories	Number of Test Data	Data Distribution
Yahoo! Answers	Question and Answer Topic Classification	10	60,000	Evenly Distributed
AG News	News Topic Classification	4	7,600	Evenly Distributed
DBPedia	Wikipedia Topic Classification	14	70,000	Evenly Distributed
IMDB	Movie Review Sentiment Classification	2	25,000	Evenly Distributed

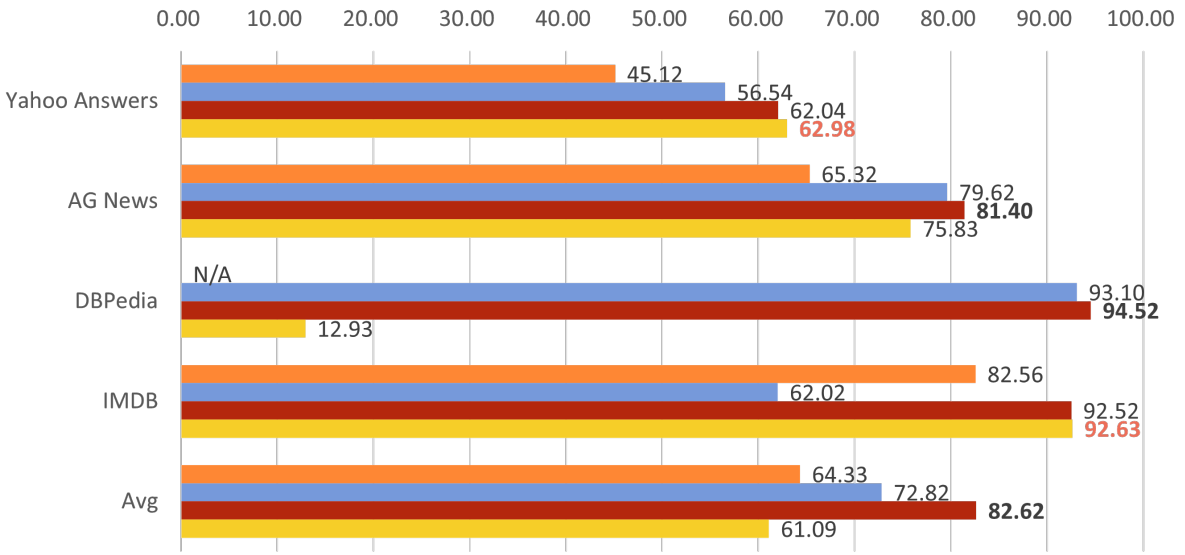
Figure 5: Overview of datasets used for evaluation

Target Category	Sports	Business & Finance	Entertainment & Music	Politics & Government
Substitute Word List (k=9) and Similarity	“Sports": 0.9999, "Water sports": 0.7659, "Air sports": 0.7651, "Whitewater sports": 0.7436, "Basketball": 0.5788, "Association football": 0.5695, "Baseball": 0.5571, "Olympic Games": 0.5134, "American football": 0.5097	"Finance": 0.8673, "Business": 0.8342, "Industry": 0.6205, "Pharmaceutical Industry": 0.5687, "Personal development": 0.5671, "Electronics companies": 0.5526, "Operations research": 0.5524, "Management": 0.5399, "Money": 0.5395	"Entainment": 0.8611, "Music": 0.8603, "Performing arts": 0.5869, "New media art": 0.5823, "Visual arts": 0.5794, "Musical groups": 0.5644, "Dance": 0.5585, "Video games": 0.5509, "Film": 0.4879	"Government": 0.8444, "Politics": 0.8334, "Political people": 0.7365, "Government agencies": 0.7239, "Public administration": 0.7052, "Politicians": 0.6373, "Criticism of religion": 0.6218, "People by legal status": 0.6036, "Social worlk": 0.5902

Figure 6: Example of Similarity between Target Categories and Substitute Words

Method	Backbone	Fine-tune dataset	Inference batch size
Prompt	GPT-3.5	--	-
TE-Wiki	BERT-base	Wikipedia, 3.387M examples	16
Self-training	DeBERTa-large	Unlabeled data from target dataset	16
UniMC	ALBERT-xxlarge	14 datasets for different tasks, 309.27k examples	16

Figure 7: Model Parameters



	Avg	IMDB	DBPedia	AG News	Yahoo Answers
Prompt tuning	64.33	82.56	0.00	65.32	45.12
TE-Wiki	72.82	62.02	93.10	79.62	56.54
Self-training	82.62	92.52	94.52	81.40	62.04
UniMC	61.09	92.63	12.93	75.83	62.98

Figure 8: Bar chart comparison of zero-shot text classification model performance

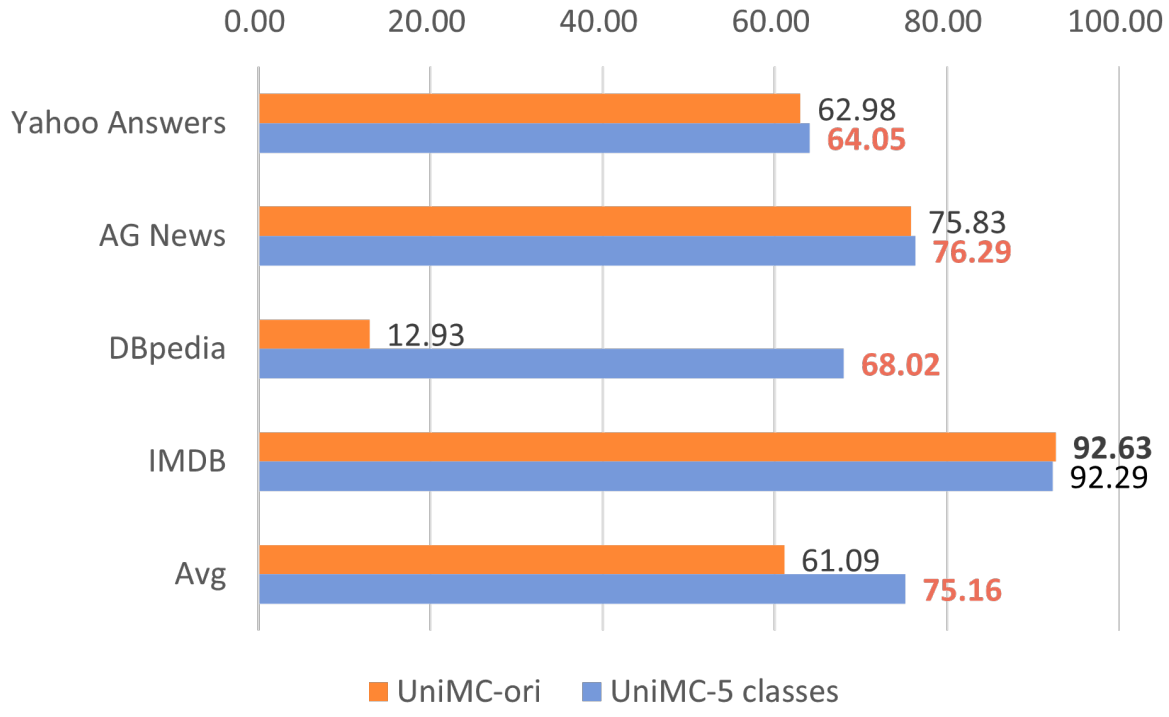


Figure 9: Bar chart comparison of zero-shot text classification model performance

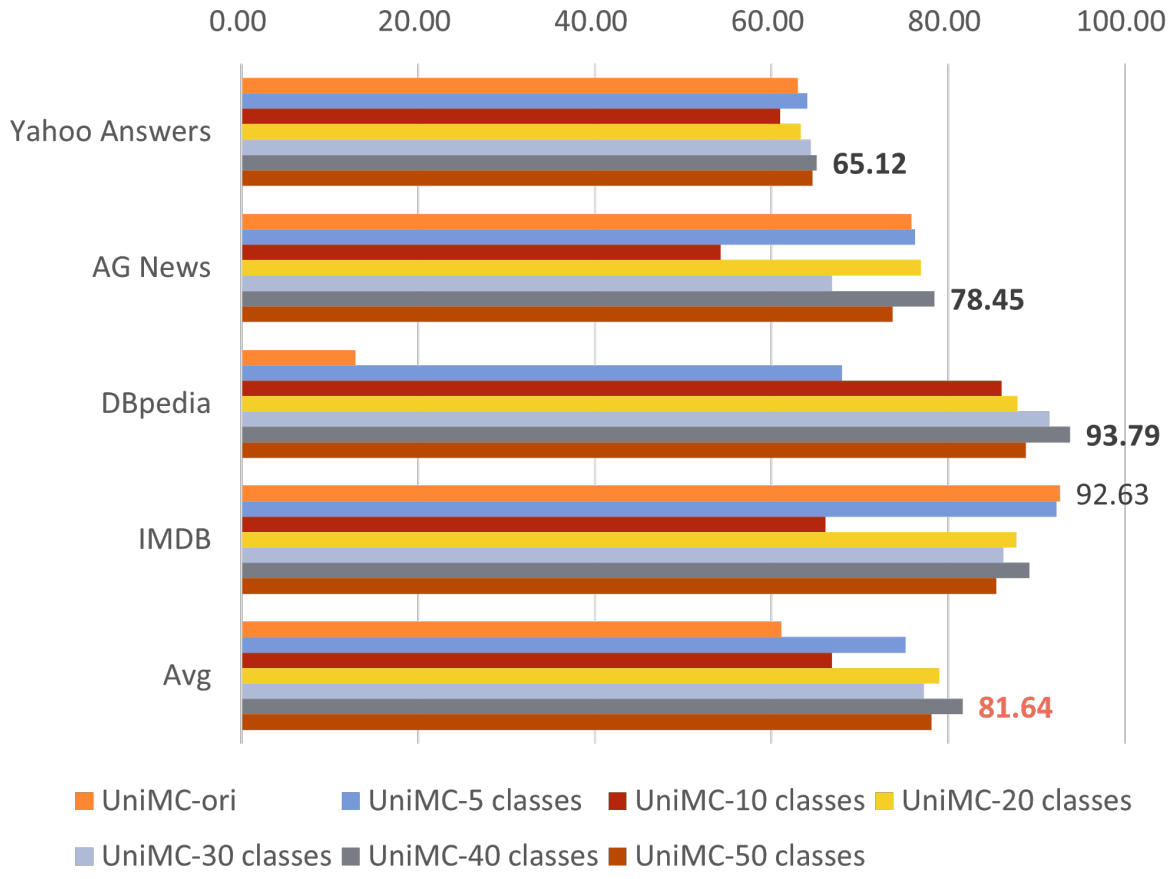


Figure 10: Bar chart comparison of zero-shot text classification model performance

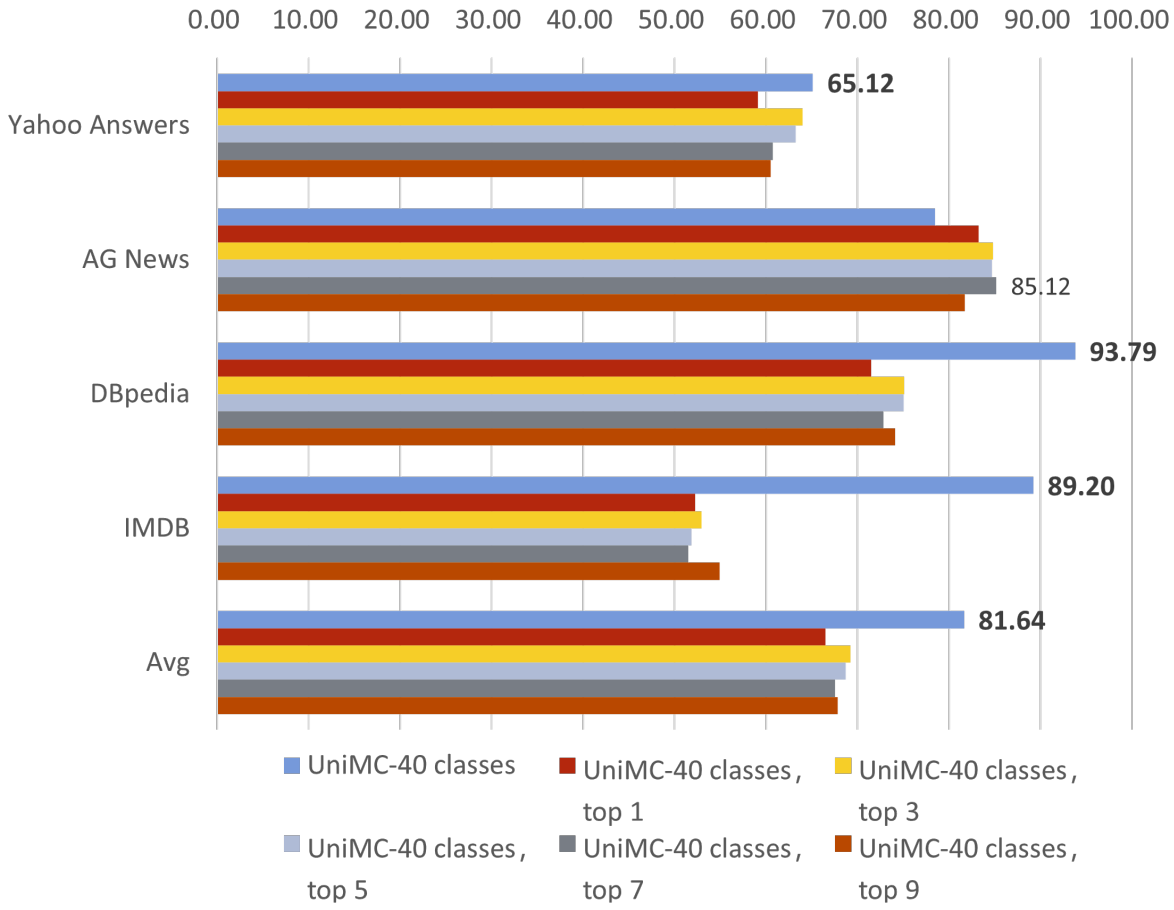


Figure 11: Bar chart of the synonym list effect exploration experiment

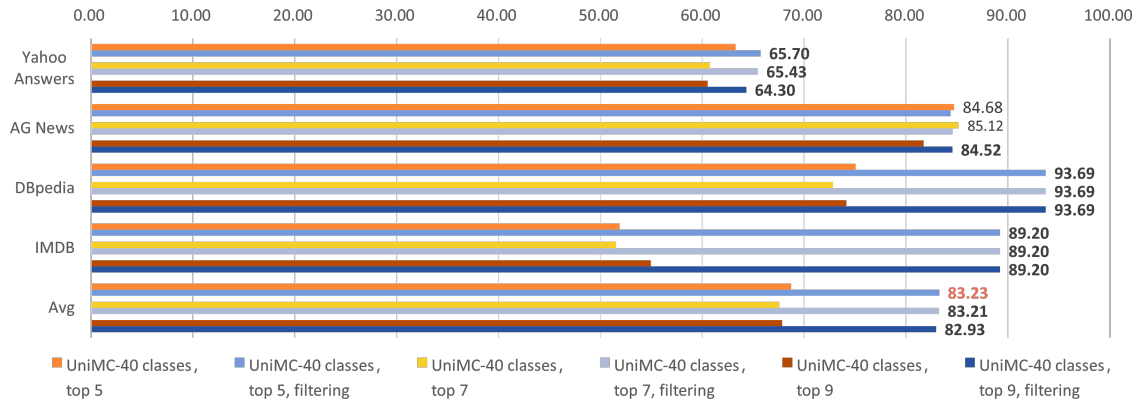


Figure 12: Bar chart of the filtering mechanism effect experiment results

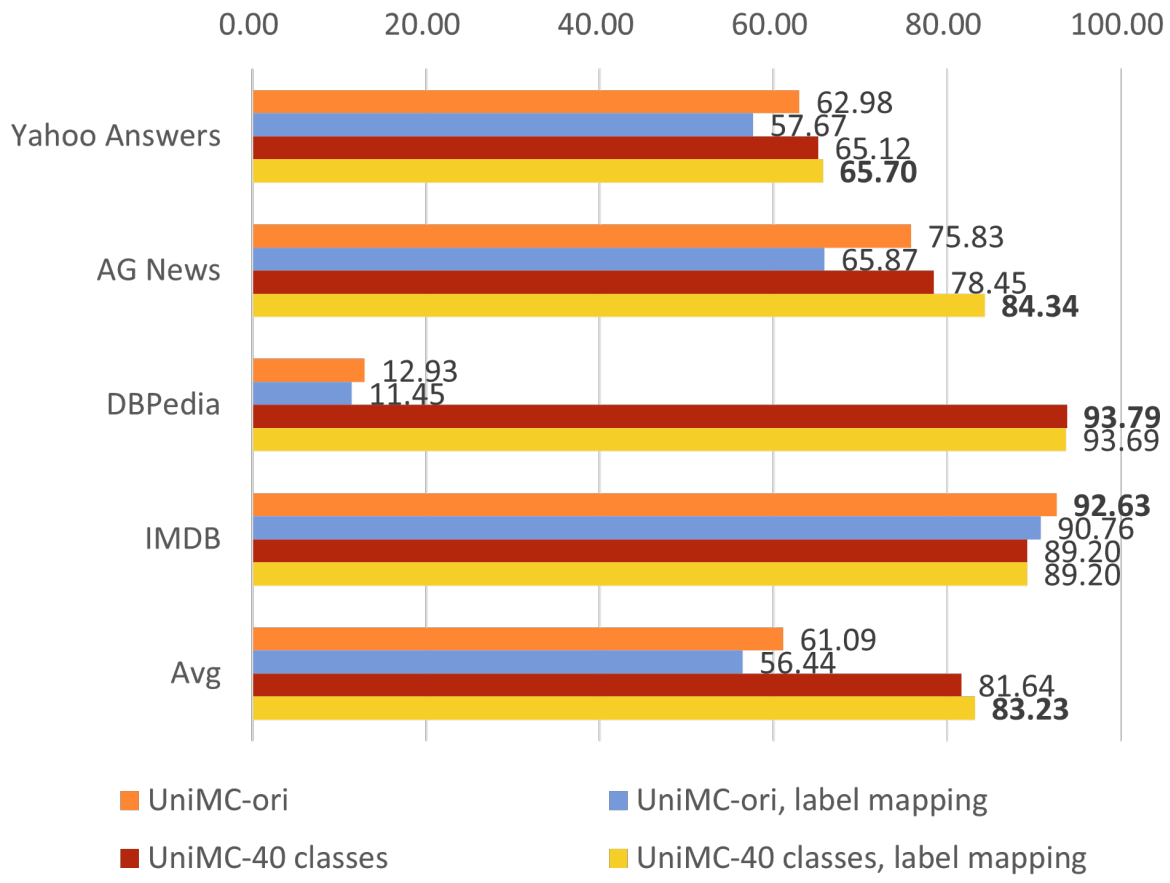


Figure 13: Bar chart of the Wikipedia fine-tuning and category mapping ablation experiment results

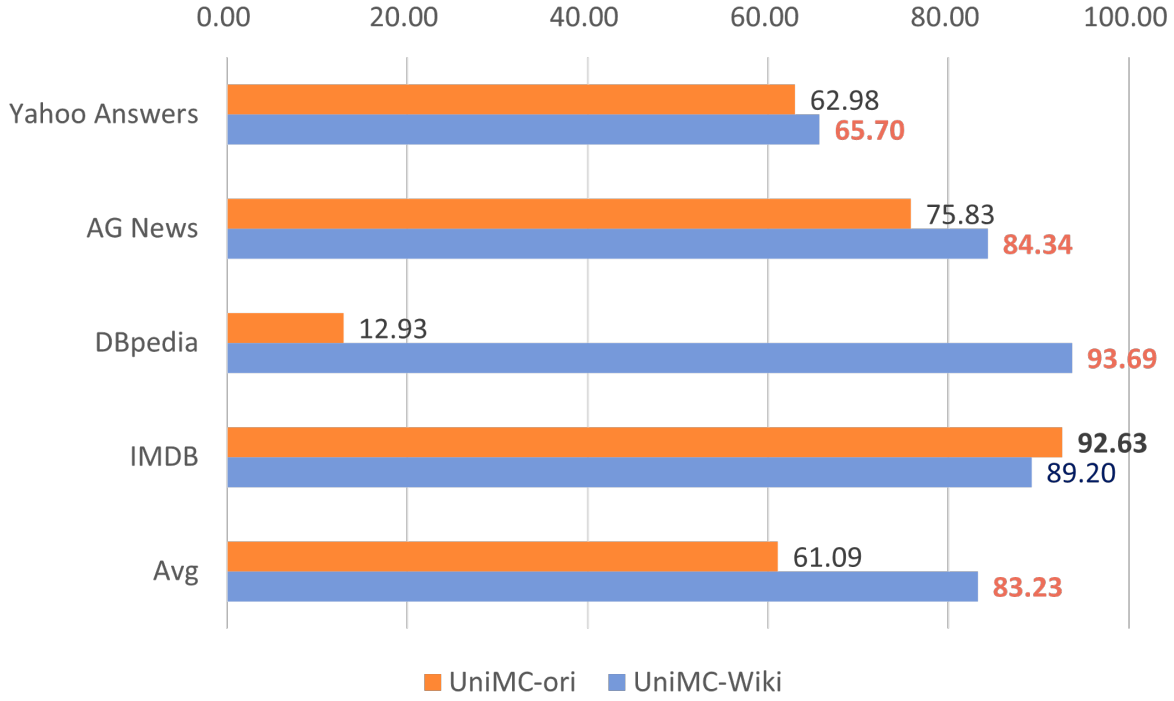


Figure 14: Bar chart of model performance comparison before and after using the research method

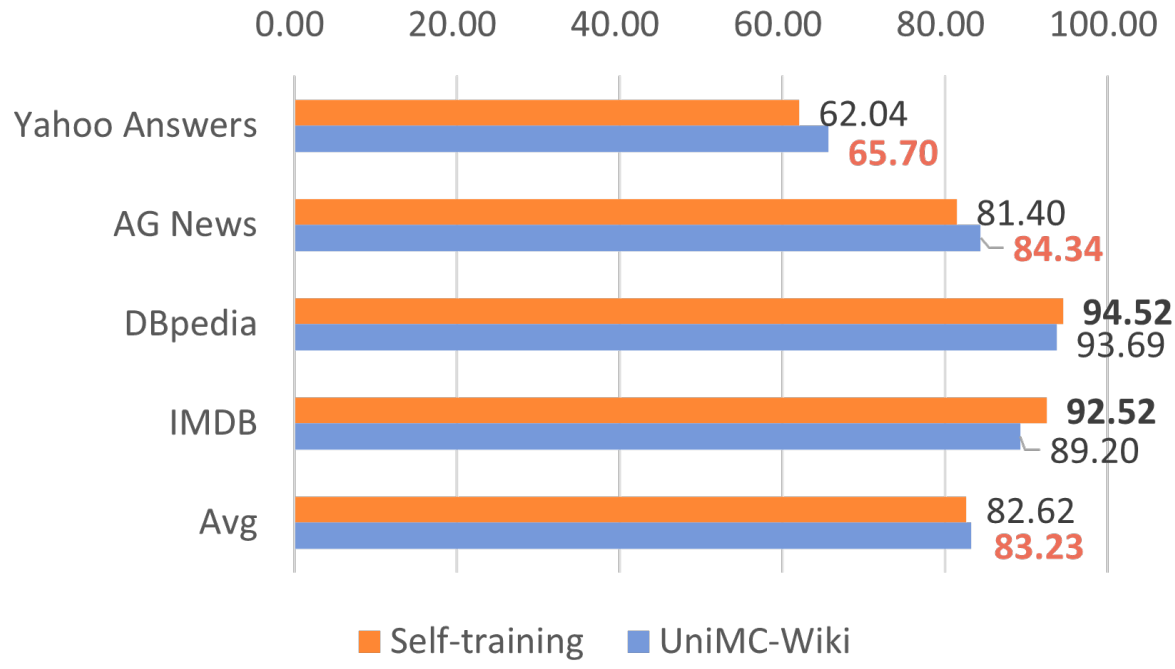


Figure 15: Bar chart of the performance comparison between UniMC-WiKi and the best model