2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Online Knowledge Distillation with Feature Disentanglement

Yifan Li[1], Zhengzhong Zhu[1], Pei Zhou[1], Kejiang Chen[1], and Jiangping Zhu[1](✉)

[1]College of Computer Science, Sichuan University, Chengdu Sichuan 610065, China
zjp16@scu.edu.cn

Abstract. Knowledge distillation is a method that trains a student model to approximate the performance of a teacher model. However, in real-world applications, the architectural discrepancy between teacher and student models often impedes the comprehensive transfer of knowledge from teacher to student. Moreover, the reduction in learnable parameters in student models poses challenges in acquiring the high-dimensional knowledge from the teacher models. due to the complexity and redundancy of the teacher model's high-dimensional features, the student model may encounter difficulties in learning these features. To address this challenge, this study proposes a knowledge distillation method based on variational autoencoders (VAE). We use VAE to compress the teacher model's high-dimensional features into low-dimensional robust features, which are extracted and transferred to the student model through the variational autoencoder loss function. Experimental results show that student models using this method achieve significant performance improvements on multiple benchmark datasets. Our research indicates that the low-dimensional robust features extracted by VAE can effectively enhance the student model's learning process, providing a new approach for knowledge distillation tasks.

Keywords: Knowledge Distillation, Variational Autoencoders, Knowledge Transfer

1 Introduction

Deep learning has been successful in various fields, including computer vision and natural language processing. However, these applications encounter significant challenges in terms of computational and storage costs. Taking natural language processing (NLP) as an example, large pre-trained language models such as BERT [1] and GPT [2] have demonstrated excellent performance. However, due to their large number of parameters, their efficient use in low-resource environments such as mobile devices and edge computing can be challenging. To promote the widespread use of deep learning applications in low-resource scenarios, knowledge distillation (KD) [3] has become a highly successful approach [3,4,5,6], which involves transferring the "knowledge" from a complex teacher model with strong learning capabilities to a simpler student model, thereby achieving model simplification. Knowledge distillation trains a smaller model,

called a student model, to imitate the performance of a larger teacher model, reducing model size while maintaining performance.

The current mainstream paradigm of distillation learning is offline distillation, where the student model learns from a pre-trained, parameter-fixed teacher model. During the distillation process, the teacher model only performs inference without updating parameters, while the student model receives fixed and unchanged knowledge from the teacher model at each training cycle. In offline distillation, large-scale teacher models do not require parameter updates, focusing solely on the student model's learning. This simplifies and makes the deployment process manageable. However, offline distillation cannot guarantee that the learning process of the teacher model matches that of the student model, nor can it adjust the knowledge distillation process of the teacher model in real-time based on the learning status of the student model.

One effective approach to dealing with the fixed parameters of large models is to use online distillation. Researchers such as [4] train both the teacher and student models simultaneously, allowing the student to learn from the optimized path of the teacher model and transform it into a competent approximator. For example, [6] and [7] introduced meta-learning principles based on online distillation to optimize the online learning process [6,7]. Moreover, [6] and [7] further took into account the generalization ability of students on their validation sets. [8] also introduces the concept of distillation impact in the area of meta distillation.

Despite the significant developments in deep learning through knowledge distillation, they all overlooked a problem, which is that they directly transmit the teacher's high-dimensional knowledge to students. However, due to the high-dimensional and complex feature representations of the teacher model, the student model faces significant challenges in learning these high-dimensional features. The complexity and redundancy of these high-dimensional features can make it difficult for the student model to effectively extract and utilize the knowledge from the teacher model, thus impacting the effectiveness of knowledge distillation. Fig. 1 illustrates our primary motivation: to streamline student learning by compressing the teacher’s high-dimensional knowledge using variational autoencoders (VAE). We can draw inspiration from the human learning process. Imagine a young student in a classroom learning mathematics. If the teacher directly introduces multiplication, the student might feel confused and overwhelmed. However, if the teacher starts with addition, thoroughly explaining the basic concepts, the student will not only grasp addition but also find it easier to understand multiplication.

This process reveals an important principle: learning low-dimensional knowledge is the foundation for understanding high-dimensional knowledge. Similarly, in knowledge distillation, the teacher model should prioritize imparting low-dimensional knowledge that the student model can easily understand and assimilate. This approach not only enhances the student model's learning effectiveness but also lays a solid foundation for it to learn high-dimensional knowledge in the future. VAE can compress high-dimensional features into low-dimensional robust features through the VAE loss function. These low-dimensional features retain the critical information of the teacher model while eliminating redundant parts of the features, thereby simplifying the learning process for the student model. In this way, the student model can more effectively

capture the useful knowledge from the teacher model, enhancing its performance in various tasks. The basic architecture of the model is shown in Fig. 2.

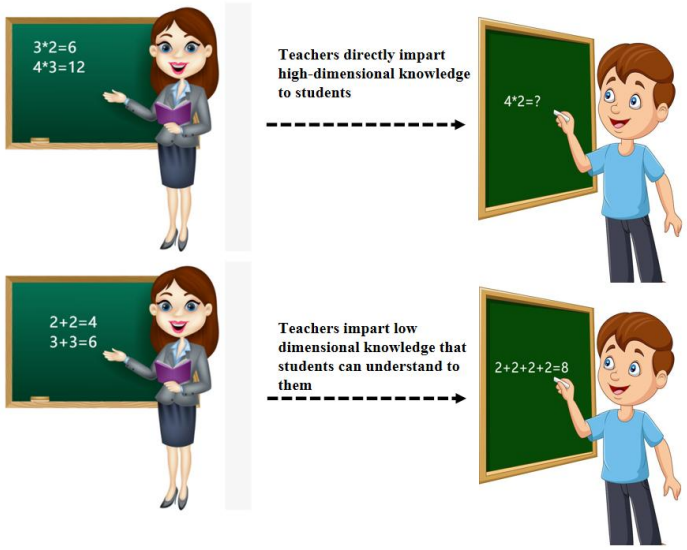

Fig. 1. The teacher model directly imparts high-dimensional knowledge to the student model, but the student model may not learn well. Therefore, the teacher model focuses on imparting low dimensional knowledge that students can learn well.

In summary, our contributions can be outlined as follows:

- Proposing a knowledge distillation method based on VAE to extract low-dimensional robust features from the teacher model and transfer them to the student model, thereby enhancing the generalization ability of the student model.
- Designing a constraint method to address the structural differences between different model architectures. This method provides a more flexible and scalable solution for knowledge distillation.
- Demonstrating the superiority of the proposed Online Knowledge Distillation Framework with Feature Disentanglement (KDFD) on 8 benchmark datasets through the comparative evaluations against the state-of-art methods in five tasks.

2 Related Work

2.1 Online Knowledge Distillation

Pretrained Language Models (PLMs) have achieved notable success in text representation [9,10]. Recent studies aim to improve PLM-based models with specific pretraining tasks [11,12,13,14]. Knowledge distillation, introduced by [3], converts large models into smaller, efficient ones while maintaining generalization power. Traditional

methods use KL divergence-based loss to align teacher and student model logits [15, 16]. Initially, offline distillation was proposed, where the teacher model is fine-tuned and frozen before transferring knowledge [3,17,18]. To address offline distillation's limitations, online distillation was introduced, allowing simultaneous teacher-student updates in an end-to-end manner [19]. Advanced methods like ProKT [4] enable students to learn from the teacher's optimization trajectory, showing how the teacher evolves from a random classifier to a strong model. However, challenges persist, such as teachers not adapting to students' capabilities [6]. MetaDistil [6] uses meta-learning and quiz data to enhance student performance and generalization, leveraging feedback on student progress to refine teacher knowledge transfer. LGTM [8] examines how distillation impacts student learning by evaluating performance on the validation set for each training sample.

2.2 Textual Feature Disentanglement

The disentanglement of latent space is first explored in the field of computer vision, and features of images (such as rotation and color) have been successfully disentangled [20]. In NLP tasks, it is used to address the decoupling of latent representations of text, such as text style and content [21], syntax and semantics [22], opinions and plots in user reviews [23], fairness representation and bias against sensitive attributes [24]. They rely on Variational Auto-Encoders or some variations [25], to restore the original feature from the space of disentanglement. In addition, there are methods to facilitate the separation of specific feature spaces by imposing regularization constraints on different tasks [26]. In this paper, inspired by the above disentangled methods, we promote the effect of cross-domain text classification by separating robust and unrobust features.

The goal of model compression algorithms is to transform large and complex pre-trained models into more streamlined and compact versions.

3 Methodology

3.1 Revisiting Knowledge Distillation

Vanilla Distillation. Suppose we have a model representing a teacher, denoted by T, and a model representing a student, denoted by S. The respective model parameters are θ_t and θ_s, i.e., $\theta_t \in R^{|t|\times 1}$ and $\theta_s \in R^{|s|\times 1}$. Given a labelled dataset D with N samples, $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$.

During the process of knowledge distillation, a properly trained teacher model is established. The parameters of the teacher model are then fixed, and its outputs are used to train a student model. The loss functions for the student network can be expressed formally as follows:

$$\mathcal{L}_s(\mathcal{D}; \theta_s; \theta_t) = \frac{1}{N}\sum_{i=1}^{N}\left[\alpha_s \mathcal{L}_{ce}\left(y_i, S(x_i; \theta_s)\right) + (1-\alpha_s)\mathcal{L}_{kd}\left(T(x_i; \theta_t), S(x_i; \theta_s)\right)\right] \quad (1)$$

The formula includes two loss functions: cross-entropy loss ($\mathcal{L}_{ce}$) and knowledge extraction loss ($\mathcal{L}_{kd}$), which is usually calculated using Kullback-Leibler (KL) divergence. α denotes the weight between the cross-entropy loss and the knowledge extraction loss. The goal of this loss function is to balance two models: enabling the student network to learn effective knowledge from the teacher.

The update of the student model is expressed as follows:

$$\theta_s^{m+1} = \theta_s^m - \eta_s \nabla \mathcal{L}_s(\mathcal{D}; \theta_s; \theta_t) \tag{2}$$

where η_s represents the learning rates for the models. The time steps before and after the model parameter updates are denoted as m and $m+1$, tracking the evolution of the model parameters during training.

However, in the vanilla distillation method, the teacher's parameters remain constant during the distillation process, limiting the teacher's ability to adjust behavior based on student feedback.

Online Distillation. To increase the flexibility and efficiency of deep learning model training, a new technique called online distillation has been proposed [19]. This approach allows the teacher model to be aware of the learning progress of the student model, and adjust its output distribution to match that of the student model:

$$\mathcal{L}_t(\mathcal{D}; \theta_s; \theta_t) = \frac{1}{N}\sum_{i=1}^{N}\left[\alpha_t \mathcal{L}_{ce}\left(y_i, T(x_i; \theta_t)\right) + (1-\alpha_t)\mathcal{L}_{kd}\left(T(x_i; \theta_t), S(x_i; \theta_s)\right)\right] \tag{3}$$

Updates have been made to the parameters of the training process for both models:

$$\theta_t^{m+1} = \theta_t^m - \eta_t \nabla \mathcal{L}_t(\mathcal{D}; \theta_t^m; \theta_s^m) \tag{4}$$

$$\theta_s^{m+1} = \theta_s^m - \eta_s \nabla \mathcal{L}_s(\mathcal{D}; \theta_s^m; \theta_t^{m+1}) \tag{5}$$

By updating the parameters, the teacher model is made aware of the optimization progress of the student model [4], resulting in significant improvements. However, it is important to note that online distillation emphasizes the mutual influence between teacher and student but does not take into account any upper limit to the capabilities of the student model. This limitation may cause the student model to lack a complete understanding of certain knowledge, which may affect its ability to generalization [27]. Therefore, when considering the practical application of online distillation, it is crucial to carefully assess the capacity limitations of the student model and explore strategies to maximize its generalization capabilities.

3.2 An Overview of Our Methodology

The approach comprises two main parts. The first part enhances information interaction between models using span constraints. The second part focuses on learning low-dimensional information through information compression representations. Fig. 2 illustrates the model structure.

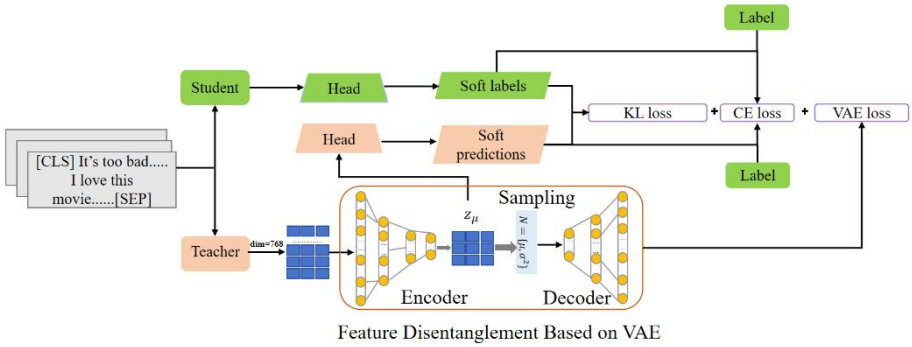

Fig. 2. Overview of the KDFD Framework. The KDFD framework initially imposes constraints on the spans between models, followed by emphasizing compressed information in the final module. The output features of the base model are fed into VAE for disentanglement. The robust feature z_{μ} is used to predict the sample labels. Finally, cross-entropy loss $\mathcal{L}_{ce}$, VAE loss $\mathcal{L}_{vae}$ and knowledge extraction loss $\mathcal{L}_{kd}$ are used to co-optimize the model.

We expect that the model can disentangle robust features in the continuous latent feature space, and it is used for effective model generalization. Inspired by some related work on textual feature disentanglement [21, 22], we adopt VAE to separate robust and unrobust features from sample feature space [25]. Specifically, we use a probabilistic latent variable z to encode the representation h, and then decode h from z.

Firstly, h_j represents the outputs of the hidden layer j, where k is the number of hidden layers. The representation of the hidden layers is as follows:

$$p(h) = \int p(z)p(h \mid z)\mathrm{d}z \tag{6}$$

$$\mathcal{L}_{vae} = -E_{q(z|h)}[\log p(h \mid z)] + KL\big(q(z \mid h)\big|p(z)\big) \tag{7}$$

where $q(z|h)$ is the posterior given by the decoder, which is formed by $N(\mu,\mathrm{diag}\,\sigma^2)$. KL is Kullback-Leibler divergence. Here, μ and σ^2 can be regarded as independent of each other under the premise of the standard normal, we present the relevant proof in the Appendix. Therefore, we use their corresponding representations to represent robust and unrobust features, instead of a simple feature split of z [22]. In practice, they can be modeled by two independent linear transformations and represented as z_{μ}, z_{σ}. Next, to ensure the robustness of z_{μ}, it should be able to help the model make correct predictions. Therefore, a classification head is used to predict the label of the current sample, from z_{μ} by cross entropy (CE):

$$\mathcal{L}_{ce} = CE\left(softmax\left(Head(z_{\mu})\right)\right) \tag{8}$$

where $Head(\cdot)$ is modeled using a linear transformation, which maps the input representations to the latent label space. By optimizing the above loss, it is possible to ensure the effectiveness of robust features for the classification task.

3.3 Training and Inference

Finally, the main body of training is the student model, so the overall loss function is the joint loss of three different loss functions:

$$\mathcal{L}_{all\_t} = \lambda \mathcal{L}_{vae}^{m} + \mathcal{L}_t(\mathcal{D}; \theta_s^m; \theta_t^m) \tag{9}$$

$$\mathcal{L}_{all\_s} = \mathcal{L}_s(\mathcal{D}; \theta_s^m; \theta_t^{m+1}) \tag{10}$$

where λ are the weighted coeffcient. Throughout the training process, all parameters of the teacher are frozen as it only provides prior knowledge of unrobust features. The following pseudocode details the training loop:

Require: student θ_s, teacher θ_t, train set D
Require: η_s, η_t: learning rate for the student and the teacher
Require: M: the maximum number of the training steps
while not done **do**
  while $m < M$ **do**
    Calculate the $\mathcal{L}_{all\_t}$
    Update $\theta_t: \theta_t^{m+1} = \theta_t^m - \eta_t \nabla_{\mathcal{L}_{all\_t}}$
    Calculate the $\mathcal{L}_{all\_s}$
    Update $\theta_s: \theta_s^{m+1} = \theta_s^m - \eta_s \nabla_{\mathcal{L}_{all\_s}}$
    $m \leftarrow m + 1$
  end while
end while

In the process of inference, the encoder part of the student model is used to predict the label of a new sample by the robust feature z_μ.

4 Experiment

4.1 Baseline Models

The study compares KDFD with eight baselines: 1) BERT-PKD [28] 2) TinyBERT [29] 3) ProKT [4] 4) TBERT-of-Theseus [30] 5) PESF-KD [31] 6) Meta Distill [6] 7) ReptileDistil [7] 8) LGTM [8].

4.2 Experimental Settings

In previous works, models like BERT-PKD [28] and MetaDistill [6] have initialized their student models by truncating certain layers of the pre-trained BERT_{base} model. Unlike the aforementioned methods, TinyBERT [29] and ReptileDistill [7] models have utilized a general TinyBERT[1] for student initialization, which learns domain-

[1] https://huggingface.co/huawei-noah/TinyBERT_General_6L_768D

agnostic knowledge from intermediate layers of the pre-trained BERT_{base} model. Therefore, we opt for the general TinyBERT_6 to initialize the student model, aiming for improved generalization capability and performance. we can obtain a teacher model with comparable performance with BERTBASE reported on the GLUE official leaderboard[2].

4.3 Training Details

We train our experimental teacher and student models simultaneously. The teacher model takes BERT and is individually tuned for each task. We set the maximum sequence length to 128 and the batch size to 32. The parameters at α_s and α_t are set to 0.4 and 0.9 respectively. Using a grid search strategy, we adjust the learning rates for the teacher model η_t from the set {1e-5, 3e-5, 5e-5} and for the student model η_s from the set {1e-5, 3e-5, 5e-5} under temperature settings of 1 and 1.5.

Moreover, to ensure the reliability of the experimental results, we conducted multiple experiments using different random seeds to validate the stability of the model's performance. By meticulously documenting various parameter settings and outcomes throughout the experimental process, we can comprehensively evaluate the effectiveness of the proposed KDFD and its applicability across different tasks.

5 Results and Analysis

5.1 Main Results

Based on the results presented in Table 1, we performed a comprehensive analysis of the performance of different methods on the GLUE dataset test set. The results show that KDFD achieved clear improvements in 6 out of 8 tasks. In particular, its performance on the smaller datasets of CoLA, MRPC, RTE, and STS-B was commendable. However, there was no significant improvement on larger datasets such as QQP and MNLI. This could be due to the large amount of data in large datasets, allowing sufficient time and resources for the learning process to capture the transfer of this low-dimensional information. Nevertheless, KDFD was still close to the optimal performance level. It showed optimal performance on the CoLA dataset, probably due to the nature of the task and the use of the Matthews correlation coefficient as an evaluation metric. When compared to online distillation methods, KDFD showed superior overall performance to methods such as ProKT and PESF-KD. This highlights the importance of transferring low-dimensional information that students can understand. Overemphasizing the comprehensive learning of teacher knowledge may lead students to over-fit teacher output, thereby reducing their generalization ability.

[2] https://gluebenchmark.com/leaderboard

Table 1. Experiment results on the test sets of GLUE. All of the listed student models have identical architectures, consisting of 6 Transformer layers, 66 million parameters, and 1.94x acceleration. The best results for each dataset are highlighted in bold, while the second-best results are underlined.

Methods	CoLA (8.5k) Mcc	SST-2 (67k) Acc	MRPC (3.7k) F1/Acc	STS-B (5.7k) Pear/Spea	QQP (364k) F1/Acc	MNLI (393k) Acc m/mm	QNLI (105k) Acc	RTE (2.5k) Acc
$BERT_{base}$(Teacher)	52.1	93.5	88.9/84.8	87.7/85.8	71.2/89.2	84.6/83.4	90.5	66.4
$TinyBERT_6$(Student)								
BERT-PKD [28]	43.5	92.0	85.0/79.9	83.4/81.6	70.7/88.9	81.5/81.0	89.0	65.5
TinyBERT [29]	51.1	93.1	87.3/82.6	85.0/83.7	71.6/89.1	84.6/83.2	90.4	70.0
ProKT [4]	-	**93.6**	88.1/83.8	-	71.2/89.2	84.2/83.4	90.9	-
BERT-of-Theseus [30]	47.8	92.2	87.6/83.2	85.6/84.1	71.6/89.3	82.4/82.1	89.6	66.2
PESF-KD [31]	-	91.5	86.0/80.6	-	70.3/88.7	81.5/80.6	87.6	65.1
Meta Distill [6]	50.7	93.5	88.7/84.7	86.1/85.0	71.1/88.9	83.8/83.2	90.2	67.2
ReptileDistil [7]	47.9	92.8	89.2/85.4	87.1/85.9	71.0/89.0	83.6/82.9	90.4	73.5
LGTM [8]	-	93.4	88.1/83.3	-	**71.7**/89.3	83.6/82.5	90.2	67.4
ICR-KD [32]	46.5	93.1	89.4/85.7	87.5/86.7	71.4/89.2	**84.7/83.6**	91.3	73.5
KDFD	**51.4**	92.4	**89.7/86.1**	**87.9/87.0**	71.2/**89.4**	84.3/**83.6**	**91.5**	**73.6**

5.2 Impact of Different Features Compress

In experimental settings, a structure known as an autoencoder (AE) is used. AE is a specialized neural network in which the input and output dimensions are equivalent [25]. Consisting of two main components, an encoder and a decoder, the encoder is used to encode the high-dimensional input z into low-dimensional latent variables h, thereby encouraging the neural network to learn the most informative features. Conversely, the decoder aims to reconstruct the hidden layer latent variables h back to the original dimensionality, ideally achieving a perfect or approximate restoration of the original input. Based on this research, we decided to use the method of capturing compressed representations of models for knowledge distillation. To achieve this, we have used AE to capture effective low-dimensional features from the teacher model and to capture the primary variance in the data. The comparison results of the AE and VAE based on the GLUE test set are shown in Table 2.

Table 2. Autoencoder and vae based on the GLUE test set.

Methods	CoLA (8.5k) Mcc	SST-2 (67k) Acc	MRPC (3.7k) F1/Acc	STS-B (5.7k) Pear/Spea	QQP (364k) F1/Acc	MNLI (393k) Acc m/mm	QNLI (105k) Acc	RTE (2.5k) Acc
AE	**43.8**	92.8	**88.3**/84.0	**86.8/85.9**	71.0/88.6	**84.0**/83.2	**91.0**	72.5
VAE	43.2	**92.9**	88.1/**84.2**	86.7/85.6	**71.4/88.8**	83.7/**83.3**	90.9	**72.8**

5.3 Ablation Analysis

To validate the effectiveness of our approach, we performed ablation experiments. The results are presented in Table 3. We analyzed the role of the variational autoencoder in model distillation (denoted as Vanilla VAE). The results of the distillation experiments show a significant improvement in performance when the VAE is used. Compared to models without the use of the VAE, the performance of models with the Variational autoencoder is more robust. This highlights the effectiveness of the VAE in facilitating the student model to capture important feature information.

Table 3. Ablation results based on the GLUE test set.

Methods	CoLA (8.5k) Mcc	SST-2 (67k) Acc	MRPC (3.7k) F1/Acc	STS-B (5.7k) Pear/Spea	QQP (364k) F1/Acc	MNLI (393k) Acc m/mm	QNLI (105k) Acc	RTE (2.5k) Acc
Vanilla KD	43.7	92.0	88.8/84.8	86.4/85.6	70.8/88.6	84.0/83.2	90.8	72.5
KDFD	**51.4**	**92.4**	**89.7/86.1**	**87.9/87.0**	**71.4/89.4**	**84.3/83.6**	**91.5**	**73.6**

5.4 Feature Correlation Analysis

To analyze the correlation between the original features and those extracted by the KDFD model, we calculate the covariance matrix for each dimension in RTE dataset. As depicted in Fig. 3, the representations generated by the KDFD model exhibit low correlations across all dimensions, indicating that KDFD facilitates the learning of a more disentangled representation.

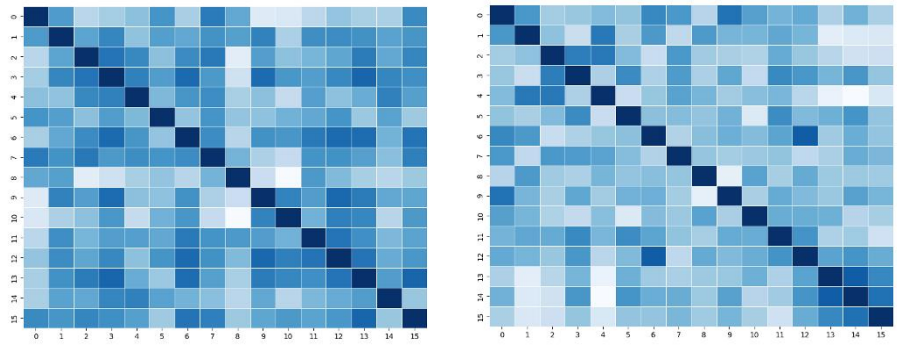

Fig. 3. The covariance matrix of the student model's representation. The darker color denotes a large correlation value. Due to space limitation, we only report the results for the first 16 dimensions.

5.5 Visualization

Further, through the visualization of the representations, we determine the impact of feature disentanglement. Specifically, tSNE is used to project 64-dimensional features into latent space. In Fig. 4, we show the visualization results, the representation z_{μ} of the VAE model is used for classification so that smooth clusters can be obtained with model optimization, which can be observed in Fig. 4, the purple clusters on the right(origin) are not as smooth and compact as left(VAE), suggesting that VAE can enhance the results of label classification.

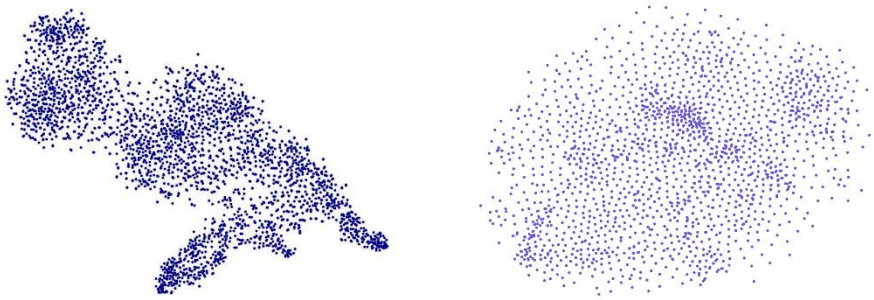

Fig. 4. Feature visualization results of z_{μ}(VAE compressed feature) and z(original features).

5.6 Impact of different compression dimensions and loss balance value

In order to verify the validity of the VAE module, several experimental groups were designed, using three different data sets and employing different compression dimensions for model distillation. The performance of these experimental groups was then compared in Fig. 5.

We first study the effects of compression dimensions by varying them in {768, 512, 256, 128,64}. The experimental results indicate that when information is emphasized by compression dimensions, the performance of the student model improves compared to the baseline experimental group. However, it is important to note that the effectiveness of this improvement can vary across different datasets, and reducing dimensions does not necessarily lead to better results. Therefore, when selecting compression dimensions, it is crucial to strike a balance between performance improvement and information loss in order to identify the optimal compression dimensions and thereby achieve the best balance between performance and efficiency.

Then we performed experiments by varying λ in {0.001, 0.002, 0.005, 0.010}. The results on four datasets are shown in Fig. 5. From them, we can see that the model performance exhibits a trend of initially increasing and then decreasing as λ increases. We can conclude that when the VAE loss function is too large, it may lead to difficulties in model convergence and overfitting issues, while when the loss function is too small, underfitting and poor feature disentanglement effects may occur. Adjusting the size of the loss function requires adjustments in hyperparameters, model architecture, and

optimizer settings to ensure that the model can learn the underlying structure of the data within an appropriate range and achieve good results in feature disentanglement tasks.

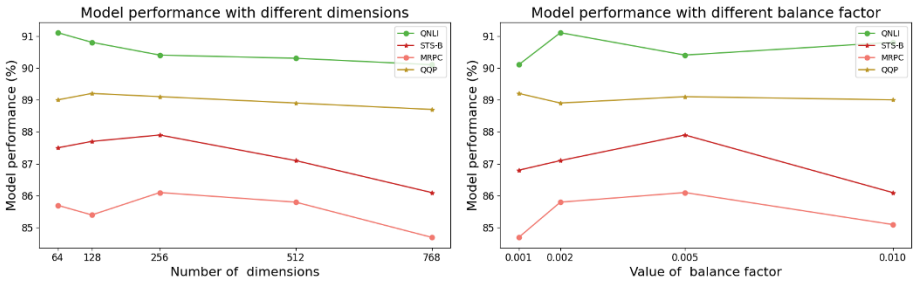

Fig. 5. FDKD Performance with different dimensions and VAE loss balance factor.

6 Conclusion

In this paper, we present a new approach for distilling knowledge in text representation called KDFD. Our technique is created to address the performance gap between large-scale pre-trained models and smaller ones. Drawing inspiration from the Information Bottleneck principle, KDFD selectively retains crucial information for the student model while discarding irrelevant information. This assists the student model in avoiding over-fitting and achieving a more disentangled representation. Through empirical experiments conducted on two text representation tasks, we demonstrate the effectiveness of KDFD in terms of accuracy and efficiency. These results establish KDFD as a promising technique for real-world applications.

References

1. J. Devlin, M.-W. Chang, K. Lee and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.
2. T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, et al., “Language models are few-shot learners,” Advances in neural information processing systems, vol. 33, 2020, pp. 1877-1901.
3. G. Hinton, O. Vinyals and J. Dean, “Distilling the knowledge in a neural network,” arXiv preprint arXiv:1503.02531, 2015.
4. W. Shi, Y. Song, H. Zhou, B. Li and L. Li, “Learning from deep model via exploring local targets,” 2020.
5. W. Park, D. Kim, Y. Lu and M. Cho, “Relational knowledge distillation,” Proc. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 3967-3976.
6. W. Zhou, C. Xu and J. McAuley, “Bert learns to teach: Knowledge distillation with meta learning,” Proc. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 7037-7049.

7. X. Ma, J. Wang, L.-C. Yu and X. Zhang, “Knowledge distillation with reptile meta-learning for pretrained language model compression,” Proc. Proceedings of the 29th International Conference on Computational Linguistics, 2022, pp. 4907-4917.
8. Y. Ren, Z. Zhong, X. Shi, Y. Zhu, C. Yuan and M. Li, “Tailoring instructions to student's learning levels boosts knowledge distillation,” arXiv preprint arXiv:2305.09651, 2023.
9. N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” arXiv preprint arXiv:1908.10084, 2019.
10. V. Karpukhin, B. Ouz, S. Min, P. Lewis, L. Wu, S. Edunov, et al., “Dense passage retrieval for open-domain question answering,” arXiv preprint arXiv:2004.04906, 2020.
11. L. Gao and J. Callan, “Unsupervised corpus aware language model pre-training for dense passage retrieval,” arXiv preprint arXiv:2108.05540, 2021.
12. Y. Zhu, X. Zhao, C. Guo and Y.-X. Wang, “Private prediction strikes back! Private kernelized nearest neighbors with individual renyi filter,” Proc. Uncertainty in Artificial Intelligence, 2023, pp. 2586-2596.
13. D. Long, Y. Zhang, G. Xu and P. Xie, “Retrieval oriented masking pre-training language model for dense passage retrieval,” arXiv preprint arXiv:2210.15133, 2022.
14. T. Shen, X. Geng, C. Tao, C. Xu, X. Huang, B. Jiao, et al., “Lexmae: Lexicon-bottlenecked pretraining for large-scale retrieval,” arXiv preprint arXiv:2208.14754, 2022.
15. S. Zagoruyko and N. Komodakis, “Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer,” arXiv preprint arXiv:1612.03928, 2016.
16. S. Sun, Z. Gan, Y. Fang, Y. Cheng, S. Wang and J. Liu, “Contrastive distillation on intermediate representations for language model compression,” Proc. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020, pp. 498-508.
17. N. Passalis and A. Tefas, “Learning deep representations with probabilistic knowledge transfer,” Proc. Computer Vision--ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI 15, 2018, pp. 283-299.
18. Z. Huang and N. Wang, “Like what you like: Knowledge distill via neuron selectivity transfer,” arXiv preprint arXiv:1707.01219, 2017.
19. Y. Zhang, T. Xiang, T.M. Hospedales and H. Lu, “Deep mutual learning,” Proc. Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4320-4328.
20. X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” Advances in neural information processing systems, vol. 29, 2016.
21. V. John, L. Mou, H. Bahuleyan and O. Vechtomova, “Disentangled representation learning for non-parallel text style transfer,” arXiv preprint arXiv:1808.04339, 2018.
22. Y. Bao, H. Zhou, S. Huang, L. Li, L. Mou, O. Vechtomova, et al., “Generating sentences from disentangled syntactic and semantic spaces,” arXiv preprint arXiv:1907.05789, 2019.
23. G. Pergola, L. Gui and Y. He, “A disentangled adversarial neural topic model for separating opinions from plots in user reviews,” arXiv preprint arXiv:2010.11384, 2020.
24. P. Colombo, G. Staerman, N. Noiry and P. Piantanida, “Learning disentangled textual representations via statistical measures of similarity,” arXiv preprint arXiv:2205.03589, 2022.
25. D.P. Kingma and M. Welling, “Auto-encoding variational bayes,” arXiv preprint arXiv:1312.6114, 2013.
26. Y. Huang, Y. Zhang, J. Chen, X. Wang and D. Yang, “Continual learning for text classification with information disentanglement based regularization,” arXiv preprint arXiv:2104.05489, 2021.

27. D.Y. Park, M.-H. Cha, D. Kim, B. Han and others, “Learning student-friendly teacher networks for knowledge distillation,” Advances in neural information processing systems, vol. 34, 2021, pp. 13292-13303.
28. S. Sun, Y. Cheng, Z. Gan and J. Liu, “Patient knowledge distillation for bert model compression,” arXiv preprint arXiv:1908.09355, 2019.
29. X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, et al., “Tinybert: Distilling bert for natural language understanding,” arXiv preprint arXiv:1909.10351, 2019.
30. C. Xu, W. Zhou, T. Ge, F. Wei and M. Zhou, “Bert-of-theseus: Compressing bert by progressive module replacing,” arXiv preprint arXiv:2002.02925, 2020.
31. J. Rao, X. Meng, L. Ding, S. Qi and D. Tao, “Parameter-efficient and student-friendly knowledge distillation,” arXiv e-prints, 2022, pp. arXiv-2205.
32. Y. Zhang, X. Zhang, J. Wang and X. Zhou, “Knowledge distillation with information compressed representations,” Proc. International Conference on Intelligent Computing, 2024, pp. 402-413.

Appendix

Suppose a set of independent samples $\{X_1, X_2, \dots, X_n\}$ follow the normal distribution $N(\mu, diag\,\sigma^2)$. The mean and variance of the samples are independent of each other.

For the above samples, the mean and variance can be expressed as: $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ and $S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2$. To prove that $\bar{X}$ and S^2 is independent of each other, an orthogonal matrix A is constructed as follow:

$$A = \begin{bmatrix} \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{\mathrm{n}}} & \cdots & \frac{1}{\sqrt{n}} & \frac{1}{\sqrt{n}} \\ \frac{1}{\sqrt{2\cdot 1}} & \frac{-1}{\sqrt{2\cdot 1}} & 0 & \cdots & 0 & 0 \\ \frac{1}{\sqrt{3\cdot 2}} & \frac{1}{\sqrt{3\cdot 2}} & \frac{-2}{\sqrt{3\cdot 2}} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \frac{1}{\sqrt{n(n-1)}} & \frac{1}{\sqrt{n(n-1)}} & \frac{1}{\sqrt{n(n-1)}} & \cdots & \frac{-1}{\sqrt{n(n-1)}} & \frac{-(n-1)}{\sqrt{n(n-1)}} \end{bmatrix} \quad (1)$$

X can be transformed into Y by the orthogonal transformation $Y = AX$, $Y = [Y_1, Y_2, \dots, Y_n]^T$, Since Y can be represented by X, the probability density function for both can be written as:

$$\begin{aligned}\mathcal{P}(Y) &= \mathcal{P}(X) = \mathcal{P}(X_1)\mathcal{P}(X_2)\dots\mathcal{P}(X_n)\\ &= \prod_{i=1}^{n}\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(X_i-\mu)^2}{2\sigma^2}}\\ &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(X_i-\mu)^2}\\ &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(X_i^2-2X_i\mu+\mu^2)}\\ &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}(\sum_{i=1}^{n}X_i^2-2n\bar{X}\mu+n\mu^2)}\end{aligned} \tag{2}$$

For Y , we have $Y^TY = (AX)^T(AX) = X^TA^TAX = X^TX$ and Y^TY can be calculated by $[Y_1, Y_2, \dots, Y_n] \times [Y_1, Y_2, \dots, Y_n]^T = \sum_{i=1}^{n} Y_i^2$, so $\sum_{i=1}^{n} Y_i^2 = \sum_{i=1}^{n} X_i^2$, Besides $Y_1 = \sqrt{\frac{1}{n}}(X_1, X_2, \dots, X_n) = \sqrt{n}\bar{X}$,So $\bar{X} = \sqrt{\frac{1}{n}}Y_1$, replace X in Eq 2 with Y , we get:

$$\begin{aligned}\mathcal{P}(Y) &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}(\sum_{i=1}^{n}Y_i^2-2\sqrt{n}Y_1\mu+n\mu^2)}\\ &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}(\sum_{i=2}^{n}Y_i^2+Y_1^2-2\sqrt{n}Y_1\mu+n\mu^2)}\\ &= (2\pi\sigma^2)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}(\sum_{i=2}^{n}Y_i^2+Y_1^2-2\sqrt{n}Y_1\mu+n\mu^2)}\\ &= \frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(Y_1-\sqrt{n}\mu)^2}{2\sigma^2}}\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{Y_2^2}{2\sigma^2}}\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{Y_3^2}{2\sigma^2}}\dots\end{aligned} \tag{3}$$

We can infer that Y is independent of each other as Eq 3 proves that the probability density function of Y can be written as the product of the density functions of its variables. Then, for S^2, we have:

$$\begin{aligned}(n-1)S^2 &= \sum_{i=1}^{n}(X_i-\bar{X})^2\\ &= \sum_{i=1}^{n}\left(X_i^2-2X_i\bar{X}+\overline{X^2}\right)\\ &= \sum_{i=1}^{n}X_i^2+\sum_{i=1}^{n}\bar{X}(\bar{X}-2X_i)\\ &= \sum_{i=1}^{n}X_i^2-n\overline{X^2}\\ &= \sum_{i=1}^{n}Y_i^2-Y_1^2\\ &= \sum_{i=2}^{n}Y_i^2\end{aligned} \tag{4}$$

Therefore, $\bar{X}$ is only affected by Y_1, while S^2 is affected by Y_2 to Y_n.