`CUE`: An Uncertainty Interpretation Framework for Text Classifiers Built on Pre-Trained Language Models (Supplementary Material)

Jiazheng Li[1] **Zhaoyue Sun**[2] **Bin Liang**[3] **Lin Gui**[1] **Yulan He**[1,2,4]

[1]Department of Informatics, King's College London, UK
[2]Department of Computer Science, University of Warwick, UK
[3]Joint Lab of HITSZ-CMS, Harbin Institute of Technology, Shenzhen, China
[4]The Alan Turing Institute, UK

1 DERIVATIONS

1.1 DECOMPOSITION OF THE PREDICTIVE UNCERTAINTY

We show how we decompose the Mean Squared Error (MSE) based predictive uncertainty into the epistemic uncertainty and the aleatoric uncertainty mentioned in §3.2.

$$\begin{aligned}\mathbb{E}\big[(y_i-\hat{y}_i)^2\big] &= \mathbb{E}\big[(y_i-\mathbb{E}[y]+\mathbb{E}[y]-\hat{y}_i)^2\big]\\ &= \mathbb{E}\big[(y_i-\mathbb{E}[y])^2\big]+\mathbb{E}[(\mathbb{E}[y]-\hat{y}_i)^2]+2\mathbb{E}[(y_i-\mathbb{E}[y])(\mathbb{E}[y]-\hat{y}_i)]\\ &= \mathbb{E}[(y_i-\mathbb{E}[y])^2]+\mathbb{E}[(\mathbb{E}[y]-\hat{y}_i)^2]+2(\mathbb{E}[y]-\mathbb{E}[y])(\mathbb{E}[y]-\mathbb{E}[\hat{y}_i])\\ &= \underbrace{\mathbb{E}[(y_i-\mathbb{E}[y])^2]}_{\text{aleatoric uncertainty}}+\underbrace{\mathbb{E}[(\mathbb{E}[y]-\hat{y}_i)^2]}_{\text{epistemic uncertainty}}\end{aligned}$$

Here, $\mathbb{E}[y]$ denotes the expectation of the ground truth label distribution. Since the first term, $\mathbb{E}[(y_i-\mathbb{E}[y])^2]$, contains the observed y_i, it can be defined as the aleatoric uncertainty. The second term represents the epistemic uncertainty since it contains the predicted $\hat{y}_i$.

1.2 INTERPRETING ENTROPY CHANGE WITH RECONSTRUCTION DIFFERENCE

In this subsection, we provide detailed derivation corresponding to the predictive entropy upper-bound mentioned in §4.1.

In our learning objective, we aim to estimate the uncertainty by adding the noise to increase the predictive entropy while keeping the classification results unchanged. For a Softmax-based classifier, the predictive uncertainty reaches the maximum when probabilities are uniformly distributed (i.e. when the predictive class probability of any of the K classes is $\frac{1}{K}$). Then, we have

$$0 \le \mathcal{H}_{\boldsymbol{e}_i}(\hat{y}_i) \le \mathcal{H}_{\boldsymbol{e}'_i}(\hat{y}'_i) \le \log K$$

Assuming that the difference between text representation $\boldsymbol{e}_i$ and the reconstructed representation $\boldsymbol{e}'_i$ is $\boldsymbol{u}$, $\boldsymbol{u}=\boldsymbol{e}'_i-\boldsymbol{e}_i$, and the prediction is obtained from the Softmax function. Let $\boldsymbol{U}$ be the maximum distance of $\boldsymbol{e}'_i-\boldsymbol{e}_i$ that causes the new $\boldsymbol{e}'_i$ confuse the classifier (i.e. when the predictive class probability equals to $\frac{1}{K}$). Then, according to the Jensen inequality, the prediction is bounded by:

$$\begin{aligned}\hat{y}'_i &\le t\cdot\text{softmax}(\boldsymbol{e}_i)+(1-t)\text{softmax}(\boldsymbol{e}_i+\boldsymbol{U})\\ &= t\cdot\text{softmax}(\boldsymbol{e}_i)+(1-t)\frac{1}{K}\end{aligned}$$

So we can get $\boldsymbol{e}_i \le \boldsymbol{e}'_i \le (\boldsymbol{e}_i+\boldsymbol{U})$, let $0\le t\le 1$ and $\mathcal{H}_{(\boldsymbol{e}_i+\boldsymbol{U})}(\hat{y}'_i)=\log K$. Considering the convexity of entropy, we

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

have:

$$
\begin{aligned}
\Delta\mathcal{H} &= \mathcal{H}_{\boldsymbol{e}_i'}(\hat{y}_i') - \mathcal{H}_{\boldsymbol{e}_i}(\hat{y}_i) \\
&\leq -(t\cdot p(\hat{y}_i) + \frac{(1-t)}{K})\text{log}(t\cdot p(\hat{y}_i) + \frac{(1-t)}{K}) + p(\hat{y}_i)\text{log}p(\hat{y}_i) \\
&\leq -t\cdot p(\hat{y}_i)\text{log}p(\hat{y}_i) - (1-t)\cdot\frac{1}{K}\text{log}(\frac{1}{K}) + p(\hat{y}_i)\text{log}p(\hat{y}_i) \\
&= (1-t)(\text{log}K - \mathcal{H}_{\boldsymbol{e}_i}(\hat{y}_i)) \\
&= \frac{||\boldsymbol{e}_i' - \boldsymbol{e}_i||^2\cdot(\text{log}K - \mathcal{H}_{\boldsymbol{e}_i}(\hat{y}_i))}{||U||^2} \\
&\propto ||\boldsymbol{e}_i' - \boldsymbol{e}_i||^2
\end{aligned}
$$

The above derivation demonstrates the generated perturbation can guarantee an upper bound of predictive entropy difference $\Delta\mathcal{H}$, and the variation of the entropy $\Delta\mathcal{H}$ is proportional to the reconstruction error $||\boldsymbol{e}_i' - \boldsymbol{e}_i||^2$, which thus can be used to interpret the predictive uncertainty.

2 UNCERTAIN FEATURE IDENTIFICATION ALGORITHM

In this section, we provide the detailed implementation of the Uncertain Feature Identification algorithm built in the `CUE` framework corresponding to §4.2. Intuitively, we use greedy search to find a locally optimal solution by identifying the most influential latent dimensions of $\boldsymbol{z}_i$ first and then estimating the influential score for each token (see in Algorithm 1). That is, we can identify input tokens that are most similar to the influential representation vector $\boldsymbol{r}_{z_i d}$ as the ones which cause predictive uncertainty by the inner product in the metric space. More concretely, assuming the PLM-encoded representation for token j is $\boldsymbol{e}_{ij}$, we can compute each token's importance score by $\text{token}^\text{j}{}_\text{score} = \langle \boldsymbol{r}_{z_i d}, \boldsymbol{e}_{ij}\rangle$. By sorting $\text{token}^\text{j}{}_\text{score}$ in descending order, we can identify input tokens that cause predictive uncertainty.

The identification of the source of the uncertainty highly relies on the influence of latent dimensions. In practice, we use a threshold α to select the most similar dimensions of $\Delta\boldsymbol{e}_i$ to construct a combination of the most influential uncertain representation $\boldsymbol{r}_{z_i D}$. The threshold α can be defined with the help of the average entropy curve and ECE histograms from the dimension importance analysis described in §5.2.

Algorithm 1 Uncertain Feature Identification

Input: original text representation $\boldsymbol{e}_i$, reconstruct text representation $\boldsymbol{e}_i'$, `CUE` decoder $\mu_{\boldsymbol{\theta}}$, token representations $\{\boldsymbol{e}_{i1}, \boldsymbol{e}_{i2}, \cdots, \boldsymbol{e}_{in}\}$, tokens $\{\text{token}_1, \text{token}_2, \cdots, \text{token}_\text{n}\}$, threshold α.

```
$\Delta\boldsymbol{e}_i = \boldsymbol{e}_i' - \boldsymbol{e}_i$
for the $d$th dimension $\boldsymbol{z}_{id}$ in $\boldsymbol{z}_i$ do
    $\boldsymbol{r}_{z_i d} = \mu_\theta(\boldsymbol{z}_{id})$
    $\text{dim}^d_\text{score} = \langle\Delta\boldsymbol{e}_i, \boldsymbol{r}_{z_i d}\rangle$
end for
for sort$(\boldsymbol{r}_{z_i d}, key = \phi(\text{dim}^d_\text{score}, \boldsymbol{r}_{z_i d}))[:\alpha]$ do
    $\boldsymbol{r}_{z_i D} += \boldsymbol{r}_{z_i d}$
end for
for $\boldsymbol{e}_{ij}$ in $\{\boldsymbol{e}_{i1}, \boldsymbol{e}_{i2}, \cdots, \boldsymbol{e}_{in}\}$ do
    $\text{token}^\text{j}{}_\text{score} = \langle\boldsymbol{r}_{z_i D}, \boldsymbol{e}_{ij}\rangle$
end for
return $\text{sort}(\text{token}_\text{j}, \text{key} = \phi(\text{token}^\text{j}{}_\text{score}, \text{token}_\text{j}))$
```

3 EXPERIMENTAL SETUP

In this section, we provide detailed dataset statistics, baseline setup, evaluation metrics and hyperparameter settings as mentioned in §5.

Datasets We evaluate our proposed framework on four datasets for *linguistic acceptability classification*, *natural language inference*, and *emotion classification*. The dataset statistics are shown in Table 1.

Datasets	CoLA	MultiNLI	Emotion	GoEmotions
Classes	2	3	6	27
Train	8,551	392,702	16,000	43,410
Dev	1,043	20,000	2,000	5,427
Test	1,043	20,000	2,000	5,426
Total	10,637	432,702	20,000	58,009

Table 1: Statistic of the datasets.

Linguistic Acceptability Classification. The CoLA (Corpus of Linguistic Acceptability) [Warstadt et al., 2018] contains sentences annotated as *grammatically acceptable* or *not*.

Natural Language Inference. The MultiNLI [Williams et al., 2018] dataset contains annotations for relations of *entailment*, *contradiction*, and *neutrality* between sentence pairs.

Emotion Classification. The GoEmotions [Demszky et al., 2020] dataset annotates Reddit comments with twenty-seven emotion labels (e.g., *fear* and *admiration*). The Emotion [Saravia et al., 2018] dataset classifies English tweets into six emotion classes (e.g., *sadness* and *joy*). Note that the GoEmotions dataset allows multi-label settings that a sentence can be annotated with more than one emotion label. In our experimental setup, we only focus on multi-class classification, and we thus filtered out those instances annotated with multiple labels in the GoEmotions.

Baselines We compare our method with the following baselines:

Label Smoothing [Gupta et al., 2021] is commonly used to deal with overfitting when using cross-entropy loss on classification tasks. It aims to uniform the distribution of labels to encourage small logit gaps and has been shown effective in calibrating PLM-based classifiers.

MC Dropout [Gal and Ghahramani, 2016] is an uncertainty estimation technique that performing multiple stochastic forward passes by randomly switching neurons off to generate ensemble of predictions. We follow the implementation of Vazhentsev et al. [2022] in our experiments.

Bayesian Neural Network (BNN) [Kabir et al., 2018] assumes weights of neural networks are random variables with a prior distribution, is thus able to obtain more robust predictions by sampling the network weights during inference, and is often used for uncertainty estimation. Motivated by Antoran et al. [2021], we also implemented a BNN plug-in framework as a comparison with our CUE framework. Specifically, we use a Bayesian linear layer[1] as the encoder and a linear layer as the decoder, and then insert them between the PLM-encoding layer and the classification layer, similar to the way we ensemble the CUE.

Evaluation Metrics We use accuracy (Acc) and macro-averaged F1 (F1) to evaluate the classification results, and Expected Calibration Error (ECE) [Desai and Durrett, 2020] calculated on predictive probabilities during inference to measure model calibration. For ECE implementation, we use the formula provided by Guo et al. [2017] as follows:

$$\text{acc}(B_m) = \frac{1}{B_m}\sum_{i\in B_m} 1(\hat{y}_i = y_i), \qquad \text{conf}(B_m) = \frac{1}{B_m}\sum_{i\in B_m} \hat{p}_i,$$

$$\text{ECE} = \sum_{m=1}^{M} \frac{B_m}{n}|\text{acc}(B_m) - \text{conf}(B_m)|$$

Predictions of n samples are grouped into M interval bins and the accuracy is calculated for each bin. B_m is the set of indices of samples that prediction confidence falls into the current interval bin. The ECE formula calculates the weighted average of the difference between the accuracy of each bin - acc(B_m) - and the average confidence - conf(B_m) - within bin B_m. In our experiments, we set $B_m = 9$.

Hyperparameters Settings We adopted the Pytorch-Transformers package[2] for the implementation of all our Transformer-based language models. For each model, we chose its corresponding base model with the following parameter size: ALBERT-

[1] `https://github.com/piEsposito/blitz-bayesian-deep-learning`

[2] `https://github.com/huggingface/pytorch-transformers`

base-v2 (11M), distilBERT-base-uncased (66M), BERT-base-uncased (110M), and RoBERTa-base (125M). We fine-tuned all these base models for 20 epochs with a batch size of 16 on each target dataset as compared to base models. For the Label Smoothing and the MC Dropout baseline, the frameworks directly modified the PLM-based models and were finetuned together with the PLM for 20 epochs with a batch size of 16. For the BNN and our CUE plug-in methods, we first fine-tuned the base models for 20 epochs and then froze the PLM encoding and classifier parameters and fine-tuned the BNN and CUE module for a further 50 epochs (with batch sizes as 16 for both modules). A learning rate of $2e-5$ and the early stop strategy have been applied to all the training. Each model has been trained 5 times with different random seeds. For each model, we report the mean and standard deviation of the evaluation results obtained by the five trained models on test sets.

4 FURTHER EXPERIMENTAL RESULTS

4.1 RESULTS WITH LATENT DIMENSION REMOVAL

As mentioned in §5.2, we study the impact of removing ranked latent variable dimensions on the other three base models: ALBERT, DistilBERT and RoBERTa. As shown in Figure 1, we can observe the same trend of ECE score and average entropy increasing when removing latent dimensions ranked by their influential scores on almost all the models while keeping accuracy and macro F1 scores almost unchanged. This proves our CUE framework can be generalized to interpret the uncertainty via latent dimensions on various models and different datasets. On the CoLA dataset, both ALBERT and RoBERTa exhibit a similar pattern compared with the BERT model; we can also observe a peak for the average entropy. The graphs show our CUE can effectively distinguish the importance between latent dimensions, and thus we can use those dimensions to interpret token level uncertainty as discussed in §4.2.

4.2 ABLATION STUDY

As mentioned at the end of the §5, we present an ablation study to investigate the contribution of various components in our framework.

Stability of training loss As discussed in §4.1, our learning objective is implemented with four loss terms. We investigate the training stability benefits from orthogonal regularisation by replacing the orthogonality loss with either a KL-divergence loss or a Wasserstein loss, where the KL-divergence loss encourages the distribution of latent variables to follow the prior standard Gaussian distribution and is widely used in general Variational Auto-encoders [Kingma and Welling, 2014, Card et al., 2018], while the Wasserstein loss enforces the latent variables to follow a Dirichlet distribution and is used in Wasserstein Auto Encoder (WAE) [Nan et al., 2019, Tolstikhin et al., 2018]. The total loss (including the reconstruction loss and the cross-entropy loss) curves during training are shown in Figure 2. We observe that the total loss replaced by either the KL-divergence loss or the Wasserstein loss exhibits fluctuation during the training process across all datasets. On the contrary, the loss with orthogonal regularisation is very stable. We further show the evaluation results with various loss terms in Table 2 [3]. It can be observed that our proposed framework with the orthogonality loss achieves better ECE results compared to using KL or Wasserstein loss.

	BERT CUE w/ Orthogonality				BERT CUE w/ KL				BERT CUE w/ Wasserstein			
Datasets	Acc	F1	$\mathcal{H}$	ECE↓	Acc	F1	$\mathcal{H}$	ECE↓	Acc	F1	$\mathcal{H}$	ECE↓
CoLA	0.8130	0.7459	0.4986	**0.0640**	0.8072	0.7300	0.3407	0.1090	0.8044	0.7240	0.3499	0.1111
GoEmotions	0.6298	0.4661	0.4333	**0.0321**	0.6298	0.4752	0.3345	0.0695	0.6263	0.4608	0.3437	0.0600
Emotion	0.9255	0.8827	0.0984	**0.0322**	0.9270	0.8847	0.0518	0.0431	0.9275	0.8853	0.0510	0.0441
MultiNLI	0.8284	0.8278	0.3650	**0.0272**	0.8294	0.8290	0.3194	0.0418	0.8291	0.8286	0.3343	0.0344

Table 2: Comparison of the performance of the BERT model fine-tuned with different loss terms on four datasets.

Training Loss Stability with Additional Loss Terms We further examine the training loss stability when adding the KL or Wasserstein distance loss terms to our framework. We fine-tuned two BERT-base uncased CUE models on the Emotions dataset. It can be observed in Figure 3 that the pairwise distance (i.e., the reconstruction loss) seems to be very unstable and keeps fluctuating during training while the orthogonality loss shows a stable decreasing trend and converges quickly. If we

[3] Results reported are single run results, which we used to generate loss stability graphs.

only compare the KL loss with the Wasserstein loss, we can see that the Wasserstein loss is more stable compared to KL. Our visualisation results show that the prior distributions assumed by the KL or the Wasserstein loss may not be suitable for reconstructing PLM-encoded representations, thus leading to higher ECE results compared to using the orthogonality constraint.

	BERT `CUE` w/ Orthogonality				BERT `CUE` w/o Orthogonality			
Datasets	Acc	F1	$\mathcal{H}$	ECE↓	Acc	F1	$\mathcal{H}$	ECE↓
CoLA	0.8123±0.0012	0.8762±0.0007	0.4991±0.0032	**0.0677**±0.0056	0.8042±0.0011	0.7230±0.0024	0.3458±0.0047	0.1121±0.0020
GoEmotions	0.6282±0.0029	0.4712±0.0087	0.4433±0.0159	**0.0326**±0.0013	0.6291±0.0019	0.4652±0.0037	0.3432±0.0014	0.0615±0.0023
Emotion	0.9259±0.0009	0.8850±0.0015	0.1031±0.0082	**0.0289**±0.0043	0.9268±0.0003	0.8848±0.0006	0.0519±0.0009	0.0430±0.0016
MultiNLI	0.8283±0.0005	0.8277±0.0005	0.3665±0.0030	**0.0262**±0.0021	0.8281±0.0009	0.8277±0.0009	0.3317±0.0009	0.0370±0.0010

Table 3: Comparison of results on BERT models trained with/without latent space orthogonality.

Latent Space Orthogonality As explained in §4.1, the orthogonality regulariser facilitates a better interpretation of the latent space. Shown in Table 3, we compare the overall performance between BERT models trained with and without latent space orthogonality. The PLMs fine-tuned with Eq. (8) outperform the counterparts without the orthogonality regularisation in ECE and average entropy on all four datasets. It is also interesting to find the f1 scores slightly decrease on the models trained without the orthogonality on almost all the datasets. Therefore, the orthogonality constraints ensure the decoder network to facilitate the same distribution on the latent space to generate reconstructed representations that lead to uncertain predictions.

We performed a further ablation study to examine the interpretability of the latent space without being regularised by orthogonality. As shown in Figure 4, without the orthogonality loss term, there is no clear relationship between the tendency of ECE scores and the average entropy during the removal of latent dimensions ranked by their influential scores. Without orthogonality we are not able to maintain the distributional consistency from the latent representation space, hence we can see an obvious fluctuation in Accuracy and F1. Without a consistent tendency, it is thus difficult to investigate each latent dimension's importance and interpret the impact of each latent dimension on model predictive uncertainty.

References

Javier Antoran, Umang Bhatt, Tameem Adel, Adrian Weller, and José Miguel Hernández-Lobato. Getting a CLUE: A method for explaining uncertainty estimates. In *ICLR*, 2021.

Dallas Card, Chenhao Tan, and Noah A. Smith. Neural models for documents with metadata. In *Proceedings of the 56th Annual Meeting of the ACL*, pages 2031–2040, July 2018. doi: 10.18653/v1/P18-1189.

Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. GoEmotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the ACL*, pages 4040–4054, July 2020. doi: 10.18653/v1/2020.acl-main.372.

Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on EMNLP*, pages 295–302, November 2020. doi: 10.18653/v1/2020.emnlp-main.21.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd ICML*, pages 1050–1059. PMLR, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th ICML*, pages 1321–1330. PMLR, 2017.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. Bert & family eat word salad: Experiments with text understanding. *AAAI*, 35(14):12946–12954, May 2021. doi: 10.1609/aaai.v35i14.17531.

H. M. Dipu Kabir, Abbas Khosravi, Mohammad Anwar Hosen, and Saeid Nahavandi. Neural network-based uncertainty quantification: A survey of methodologies and applications. *IEEE Access*, 6:36218–36234, 2018. doi: 10.1109/ACCESS.2018.2836917.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd ICLR, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Feng Nan, Ran Ding, Ramesh Nallapati, and Bing Xiang. Topic modeling with Wasserstein autoencoders. In *Proceedings of the 57th Annual Meeting of the ACL*, pages 6345–6381, July 2019. doi: 10.18653/v1/P19-1640.

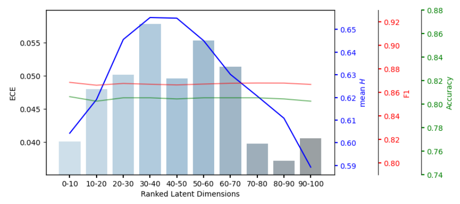
Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. CARER: Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on EMNLP*, pages 3687–3697, October-November 2018. doi: 10.18653/v1/D18-1404.

Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *ICLR*, 2018.

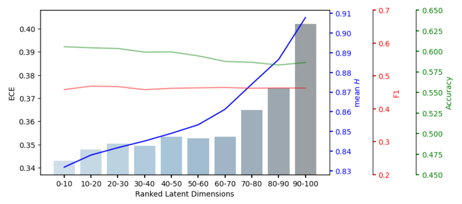
Artem Vazhentsev, Gleb Kuzmin, Artem Shelmanov, Akim Tsvigun, Evgenii Tsymbalov, Kirill Fedyanin, Maxim Panov, Alexander Panchenko, Gleb Gusev, Mikhail Burtsev, Manvel Avetisian, and Leonid Zhukov. Uncertainty estimation of transformer predictions for misclassification detection. In *Proceedings of the 60th Annual Meeting of the ACL*, pages 8237–8252, May 2022.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments, 2018. URL `https://arxiv.org/abs/1805.12471`.

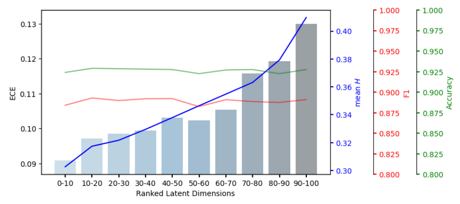
Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of NAACL*, pages 1112–1122, June 2018. doi: 10.18653/v1/N18-1101.



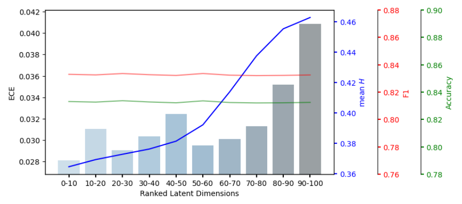
(a) ALBERT `CUE` on CoLA.



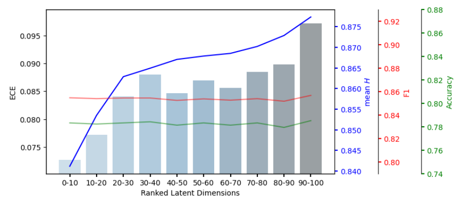
(b) ALBERT `CUE` on GoEmotions.



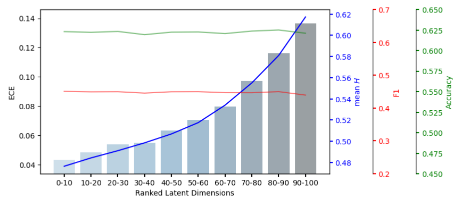
(c) ALBERT `CUE` on Emotion.



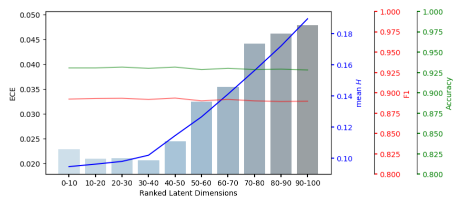
(d) ALBERT `CUE` on MultiNLI.



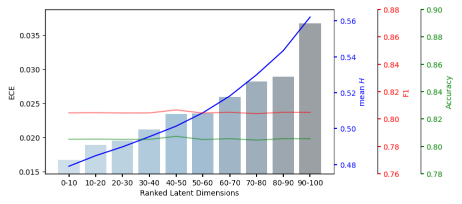
(e) DistilBERT `CUE` on CoLA.



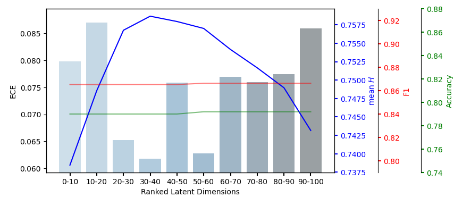
(f) DistilBERT `CUE` on GoEmotions.



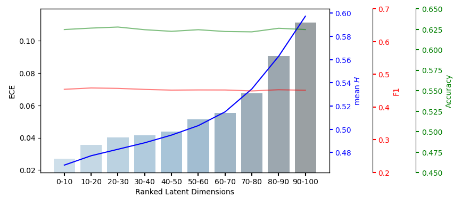
(g) DistilBERT `CUE` on Emotion.



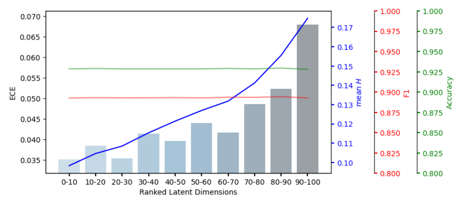
(h) DistilBERT `CUE` on MultiNLI.



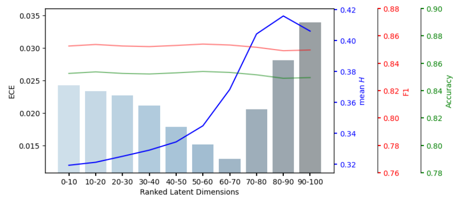
(i) RoBERTa `CUE` on CoLA.



(j) RoBERTa `CUE` on GoEmotions.



(k) RoBERTa `CUE` on Emotion.



(l) RoBERTa `CUE` on MultiNLI.

Figure 1: Evaluation results by removing latent dimensions. The x-axis represents the index of **removed** dimensions ranked by their relevance to $\Delta \boldsymbol{e}_i$, smaller index number indicates the latent dimension is more similar. Histograms show the ECE scores after removing the corresponding latent dimensions. The blue curve shows the predictive entropy. The green and red curves show classification accuracy and F1, respectively.

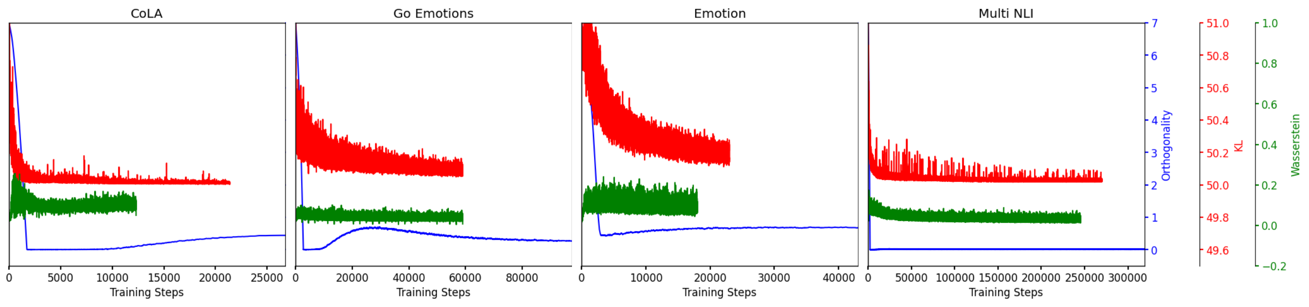
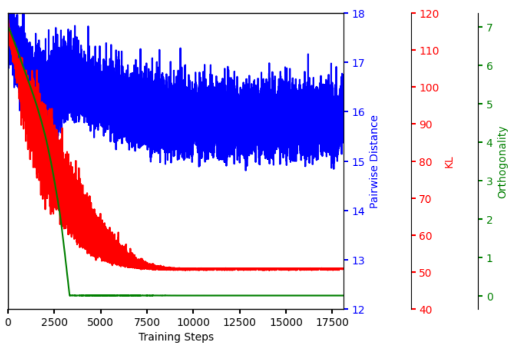
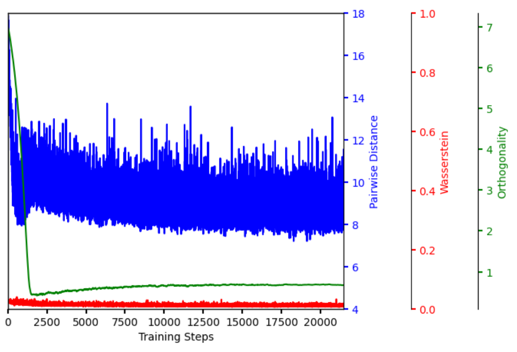


Figure 2: Comparison of the stability of the total loss for three loss terms trained with BERT model on four datasets. Red: total loss with KL divergence loss; Green: total loss with Wasserstein loss; Blue: total loss with Orthogonality loss.



(a) Additional KL divergence loss term.



(b) Additional Wasserstein loss term.

Figure 3: Comparison of BERT models trained on Emotions with additional loss term. Blue: Reconstruction loss; Red: KL loss in (a) and Wasserstein loss in (b); Green: Orthogonality loss.

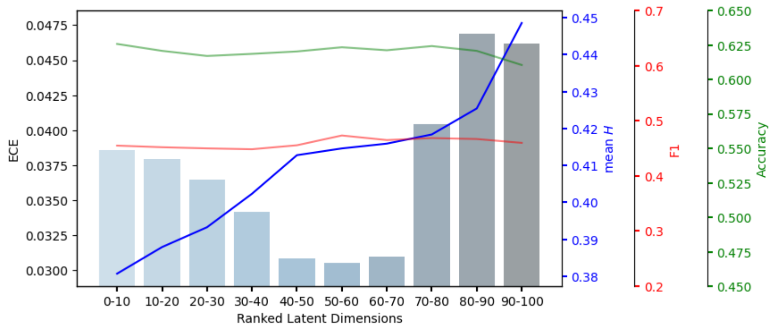


Figure 4: ECE and average entropy with latent dimension removal from the model trained without the orthogonality regulariser. The x-axis represents the index of **removed** dimensions ranked by their relevance to Δe_i, smaller index number indicates the latent dimension is more similar. Histograms show the ECE scores after removing the corresponding latent dimensions. The blue curve shows the predictive entropy. The green and red curves show the classification accuracy and F1, respectively.