Two Birds One Stone: Dynamic Ensemble for OOD Intent Classification

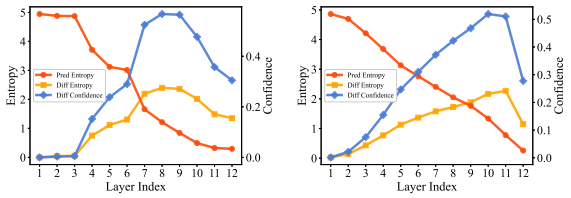
Yunhua Zhou, Jianqiang Yang, Pengyu Wang, Xipeng Qiu*
School of Computer Science, Fudan University
{zhouyh20,xpqiu}@fudan.edu.cn
yjqiang1@outlook.com
pywang22@m.fudan.edu.cn

Abstract

Out-of-domain (OOD) intent classification is an active field of natural language understanding, which is of great practical significance for intelligent devices such as the Task-Oriented Dialogue System. It mainly contains two challenges: it requires the model to know *what it knows* and *what it does not know*. This paper investigates "overthinking" in the open-world scenario and its impact on OOD intent classification. Inspired by this, we propose a two-birds-one-stone method, which allows the model to decide whether to make a decision on OOD classification early during inference and can ensure accuracy and accelerate inference. At the same time, to adapt to the behavior of dynamic inference, we also propose a training method based on ensemble methods. In addition to bringing certain theoretical insights, we also conduct detailed experiments on three real-world intent datasets. Compared with the previous baselines, our method can not only improve inference speed, but also achieve significant performance improvements. Code is publicly available.[1]

1 Introduction

With the increasing popularity of intelligent devices such as the Task-Oriented Dialogue System and the increasingly open environment they face, it becomes more and more challenging to accurately understand the intents behind the utterance of the users, on one hand, it needs to ensure the accuracy of In-domain (IND) intents, and on the other hand, it needs to be able to effectively identify OOD intent. These days, it has became a de facto standard to take the semantic representation of the last layer of finetuned Pre-train Models (PTMs) for intent understanding (Zhang et al., 2021b; Zhou et al., 2022). However, in the closed-world scenario (i.e., training and test set come from the same distribution), previous studies have pointed out that PTMs would be "overthinking" the semantic features of the sample (Kaya et al., 2019; Zhou et al., 2020), i.e., the representation of the sample may become too complex after reaching the last layer through multiple stacked transformer layers, thus affecting the final decision of the model.



(a) Overthinking in BERT (b) Overthinking in ALBERT

Figure 1: Illustration of overthinking in the open-world scenario. The model becomes more "confident" (Pred Entropy decreases with the number of layers.), but the ability to distinguish IND and OOD decreases (the difference is getting smaller). The results are obtained with the test set of CLINC-SMALL.

Naturally, in the open-world scenario (i.e., the condition of the same distribution is not maintained), will the PTMs be "overthinking" the difference between IND and OOD intents? We want to explore the evolution of the model's ability to distinguish IND and OOD through the forward process of two widely used models BERT (Devlin et al., 2019) and ALBERT (Lan et al., 2020).

To this end, we first attach extra Internal Classifiers (ICs), which also are referred to as *off-ramps*, between different internal transformer layers of the model as shown in Figure 2 (a) and train these internal classifiers and all transformer layers in two stages as suggested in Xin et al. (2020). The outputs of these classifiers are used to measure the ability of different layers to distinguish between IND and OOD, specifically, by comparing the confidence (calculated by *maximum softmax probabil-*

*Corresponding author.
[1]https://github.com/zyh190507/Dynamic-Ensemble-for-OOD

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 10659–10673
July 9-14, 2023

ity) or uncertainty (calculated by *entropy*) outputted by each layer on In- and Out-of-domain samples. Ideally, IND should be given higher confidence (or lower uncertainty) than OOD, and a larger difference or gap suggests a stronger ability to distinguish between them.

The results are shown in Figure 1. There are two types of curves in Figure 1, where the Pred Entropy (red) refers to the entropy (average value) of the sample output by the internal layers of the model, which is used to measure the certainty in its output. The curve of "Diff" shows the trend of the variation in the difference between the average values of the confidence (blue, and the orange refers to entropy) of all IND samples and OOD samples as the number of layers increases. We observe that with the increase of the number of layers, the model becomes more "confident" (lower entropy in output), but the ability to distinguish IND and OOD decreases (the difference is getting smaller) after reaching a certain peak.

The above experiments show that in the open-world scenario, PTMs also would be "overthinking" for the difference between IND and OOD. The "overthinking" behavior of the model in different scenarios also inspires us it is unnecessary to forward the last layer to distinguish IND and OOD or identify a specific class of IND during inference. To this end, we design an inference strategy to let the passed ICs **dynamic ensemble** and let these ICs vote to determine whether the sample is IND or OOD and whether they have enough confidence to exit early (Section 3.2). Furthermore, we introduce a training method that can adapt to the accurate and accelerated behavior in the inference stage. On the one hand, according to the theory of ensemble methods, we can improve the accuracy of IND recognition of the ensembled ICs by reducing the redundancy between ICs to increase the diversity among ICs during the training process. On the other hand, we can reduce the *open space risk* by regulating the recognition behavior of the adjacent space of each training sample during the training process to improve the OOD detection capability.

In addition to the theoretical explanation of the reason for introducing the training method, we also carry out detailed verification on three challenging intent datasets. The experiment shows that our method can not only substantially improve the accuracy, but also effectively accelerate inference. We summarize our key insights and contributions as follows: Firstly, we explore the "overthinking" phenomenon of the model in the open-world scenario and its impact on the discrimination between IND and OOD. We hope that these explorations can inspire follow-up studies. Secondly, we design a simple but effective inference strategy to improve the model recognition ability and speed up the model inference. At the same time, we also introduce a novel training method to adapt the inference strategy. Thirdly, detailed experiments show that compared with the existing methods, our proposed method can substantially improve the accuracy but also effectively accelerate inference.

2 Related Work

Our work is related to two types of research: **OOD Intent Classification** and **Early Exiting**.

OOD Intent Classification is also known as OOD detection. Such research can be roughly divided into two categories according to whether additional OOD samples are introduced (or synthesized) in the training process: supervised (The training set contains additional OOD samples) and unsupervised (there are no additional OOD samples in the training set). Since supervised methods require a large number of time-consuming OOD samples and the constructed OOD samples cannot cover all OOD classes, this work focuses on unsupervised methods.

Scheirer et al. (2012) articulate open space risk as the optimization objective of the task through establishing a constrained optimization. OpenMax (Bendale and Boult, 2016) models the output of the penultimate layer of the network as a specific distribution (**Weibull** Distribution) for OOD detection. MSP (Hendrycks and Gimpel, 2017) uses the confidence (maximum softmax probability) of the network to distinguish IND and OOD. DOC (Shu et al., 2017) connects N binary classifiers at the end of the network for OOD classification. LCML (Lin and Xu, 2019) introduces Margin loss to learn more discriminative semantic features to facilitate OOD detection. ADB (Zhang et al., 2021b) reduces open space risk by introducing adaptive boundaries for each IND class. Lin et al. (2021) extract image features by virtue of projecting the complexity of the images onto the network layers to construct an OOD detector in the domain of computer vision. Xu et al. (2021) use all the intermediate representations of pre-trained transformers to build an OOD detector. SCL (Zeng et al., 2021) learns

to adapt to downstream detection features by contrastive learning paradigms. KNN-CL (Zhou et al., 2020) further improves the learned semantic features by constraining positive and negative samples in contrastive learning. Different from the existing work, our method can dynamically decide whether to exit early without going through the whole model during inference, which speeds up the speed of inference and effectively guarantees the accuracy of recognition.

Early Exiting It is considered an effective way to accelerate the speed of inference of large pre-trained language models (Xin et al., 2020; Zhou et al., 2020; Liu et al., 2020; Sun et al., 2021). The mechanism allows samples to exit from some internal layers of the model in advance to accelerate the inference process of the model without losing accuracy as much as possible. However, the existing research on Early Exiting is based on the closed-world scenario. In this work, based on further elaborating the manifestations of the "overthinking" phenomenon in the open-world scenario, we introduce the Early Exiting mechanism into the open-world scenario to help improve the model's ability to distinguish IND and OOD while accelerating the inference of the model.

3 Proposed Method

3.1 Training Objective

We attach an internal classifier to each layer of BERT to enable early exiting. Denote Z_l as the output distribution generated by the Internal Classifier (IC) in the l-th layer (lowercase z_l to represent the feature of a specific input sample at layer l), $Z_{1:l}$ as the joint output of internal classifier from the first layer to the l-th layer, ϕ as Inference (ensemble) strategy.

Training ICs as an Ensemble Before that, we need to solve a concern. Since the training set only has data with IND samples, *will be trained on this training set to lead to the model overfit to the training data and damage the detection of OOD*? Fortunately, Vaze et al. (2021) show that it will not damage or even improve the recognition of IND, and even help detect OOD.

Therefore, to accelerate inference without losing accuracy, we only need to ensure the decision made by the ensemble constructed by the passed l ICs can be accurate enough in IND samples, i.e., our objective is to optimize: **min** $p(\phi(Z_{1:l}) \neq Y)$, Y is the ground truth label. Based on previous studies (Fano, 1961; Hellman and Raviv, 1970), the decision error has the following upper bound:

$$p(\phi(Z_{1:l}) \neq Y) \leq \frac{Ent(Y) - I(Z_{1:l}; Y)}{2}, \quad (1)$$

where $Ent(Y)$ is the entropy of random variable Y. $I(Z_{1:l}; Y)$ indicates mutual information between $Z_{1:l}$ and Y. According to the upper bound (1), we can optimize decision error by maximizing I. However, due to the exponential combinatorial property of possible value Z and Y, it is extremely difficult to calculate $I(Z_{1:l}; Y)$ directly. We decompose $I(Z_{1:l}; Y)$ to obtain a computable lower bound as suggested in Zhou and Li (2010); Sun et al. (2021):

$$I(Z_{1:l}; Y) \geq \underbrace{\sum_{i=1}^{l} I(Z_i; Y)}_{\mathcal{L}_{rel}} \underbrace{- \sum_{i=2}^{l} I(Z_i; Z_{1:i-1})}_{\mathcal{L}_{red}}, \quad (2)$$

where $\sum_{i=1}^{l} I(Z_i; Y)$ represents the sum of mutual information between Z_i and Y, referred to *relevancy*, which is utilized to establish the bound of the internal classifier. $\sum_{i=2}^{l} I(Z_i; Z_{1:i-1})$ is defined as *redundancy*, assessing the interdependence among classifiers. A diminished value indicates an elevated scope of dissimilitude among classifiers.

Following Sun et al. (2021), *relevancy*($\mathcal{L}_{rel}$) and *redundancy*($\mathcal{L}_{red}$) can be defined as $-\mathcal{L}_{ce}(z_i, y)$ and $-\min_{j<i} \mathcal{L}_{ce}(z_i, z_j)$ respectively. Therefore, the objective to be optimized as follows:

$$\mathcal{L}_z = -(\mathcal{L}_{rel} - \mathcal{L}_{red}) \quad (3)$$

$$= \alpha \sum_{i=1}^{L} \mathcal{L}_{ce}(z_i, y) - \beta \sum_{i=2}^{L} \min_{j:j<i} \mathcal{L}_{ce}(z_i, z_j), \quad (4)$$

where $\mathcal{L}_{ce}$ is standard cross-entropy loss. z_i is the internal representation of the input in layer i, and y is its real label. The α and β represent hyperparameters for optimization.

Reduce Open Space Risk There is no prior knowledge about OOD as mentioned earlier, we cannot directly optimize the objective of OOD. Previous research (Scheirer et al., 2012; Zhou et al., 2022) has proved that the ability of the model to detect OOD can be also improved indirectly by reducing the open space risk. Therefore, in addition to the above effective training paradigms, we also propose a way to reduce open space risk. Following Scheirer et al. (2012); Bendale and Boult (2016), open space risk can be defined as:

$$\mathcal{R}_{\mathcal{O}}(f) = \frac{\int_{\mathcal{O}} f(x)dx}{\int_{\mathcal{S}} f(x)dx}, \quad (5)$$

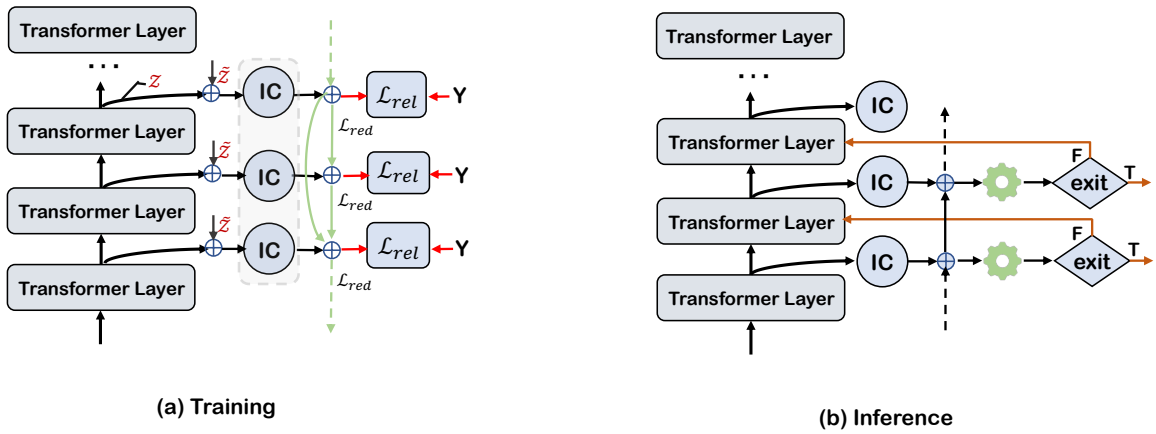


Figure 2: Overview of our proposed method. (a) The training objective consists of two parts of losses: *Relevancy loss* (red line), is used to give the bound of the internal classifiers. *Redundancy loss* (green line), is used to measure the interdependence between classifiers. (b) During inference, the ensemble composed of the passed classifiers jointly votes to decide whether to exit early or continue to move forward. See text for details.

where $\mathcal{O}$ is the open space, f is a measurable function that $f(x) = 1$ (or $f(x) > 1$) for IND intents, otherwise $f(x) = 0$ (or $f(x) <= 0$), $\mathcal{S}$ is whole semantic space containing the open space $\mathcal{O}$.

Using training samples (IND samples) $\mathcal{X}$, open space (Bendale and Boult, 2016) can be further defined as $\mathcal{O} = \mathcal{S} - \bigcup_{z \in \mathcal{X}} \sigma(z)$, $\sigma(z)$ is the minimum semantic space around the feature z.

$$\begin{aligned}\mathcal{R}_{\mathcal{O}}(f)(\downarrow) &= \frac{\int_{\mathcal{O}} f(x)dx}{\int_{\mathcal{S}} f(x)dx} = \frac{\int_{\mathcal{S}-\bigcup\sigma(z)} f(x)dx}{\int_{\mathcal{S}} f(x)dx} \\ &= 1 - \frac{\int_{\bigcup\sigma(z)} f(x)dx}{\int_{\mathcal{S}} f(x)dx} \\ &= 1 - \sum_{z}^{\mathcal{X}} \frac{\int_{\sigma(z)} f(x)dx(\uparrow)}{\int_{\mathcal{S}} f(x)dx}\end{aligned} \tag{6}$$

According to Eq.(6), to reduce the open space risk, only need to increase $\int_{\sigma(z)} f(x)dx$. Intuitively, according to the definition of f, we only need to make the semantic space around the input feature z be recognized as IND as far as possible. However, even in the infinitesimal space enclosing z, there exist a boundless number of sample points. As such, we select representative sample $\tilde{z}$ that are hard to be differentiated by the classifier to make their labels consistent with the label of z (i.e., recognized as IND). Inspired by adversarial examples (Zhu et al., 2020), $\tilde{z}$ is satisfied the following requirements:

$$\tilde{z} = z + \tau^*, \tau^* = \underset{\|\tau\| \leq \varepsilon}{\arg\max}\, \mathcal{L}_{z+\tau}. \tag{7}$$

Final finetune Objective By bringing $\tilde{z}$ into Eq.(4), we can get the loss $\mathcal{L}_{\tilde{z}}$:

$$\mathcal{L}_{\tilde{z}} = \tilde{\alpha} \sum_{i=1}^{L} \mathcal{L}_{ce}(\tilde{z}_i, y) - \tilde{\beta} \sum_{i=2}^{L} \min_{j<i} \mathcal{L}_{ce}(\tilde{z}_i, \tilde{z}_j). \tag{8}$$

The **final finetune objective** as follows:

$$\mathcal{L}_{obj} = \mathcal{L}_z + \mathcal{L}_{\tilde{z}}. \tag{9}$$

An overview of the whole training process can refer to Figure 2(a).

3.2 Inference Strategy

During inference, each internal classifier through which the input sample x_i passed will give a corresponding prediction, the prediction of the l-th internal classifier $\hat{Y}_l$ as follows:

$$\hat{Y}_l = \begin{cases} \text{OOD}, & \mathcal{G}(z_i^l) < \theta_l, \\ \text{argmax}_{k \in [K]} \mathcal{H}_k(z_i^l), & \mathcal{G}(z_i^l) \geq \theta_l. \end{cases}, \tag{10}$$

where $\mathcal{G}$ is the scoring function (**LOF** (Breunig et al., 2000) is used in this paper) whose value is used to distinguish IND (>= θ_l) and IND (< θ_l). z_i^l is the feature of x_i as layer l. $\mathcal{H}$ is softmax function. K is the number of IND intents.

The final decision is determined by the joint vote of all the internal classifiers which the sample has passed through. Only when a certain class (OOD or specific IND class) reaches or exceeds the preset threshold, the model will specify the class as the final result and early exit. At l-th layer, the inference strategy generalizing Sun et al. (2021) is

as follows:

$$\phi(\hat{Y}_{1:l}) = \frac{\max_{k \in H} \left\{ \sum_{j=1}^{l} \mathbb{I}(\hat{Y}_j = y_k) \right\}}{l^{\gamma}}, \quad (11)$$

where H is the set including IND categories and OOD (H=$[K] \cup \{\text{OOD}\}$), $\gamma \in [0, 1)$ is a hyper-parameter. When $\phi(\hat{Y}_{1:l})$ is greater than a certain threshold, the sample exits at the current l-th layer and is given an IND label (i.e., k) or recognized as OOD. See Figure 2(b) for the inference.

4 Experiments

4.1 Datasets

To verify the effectiveness and universality of our proposed method, we conducted exhaustive experiments on three widely used intent datasets. These datasets are summarized as follows:

CLINC-FULL (Larson et al., 2019) is a very popular dataset, which encompasses a broad range of intents, totaling 150 across 10 domains. The entire dataset consists of 22500 in-domain samples and 1200 Out-of-domain samples.

BANKING (Casanueva et al., 2020) is a kind of dataset about the banking business, with 77 categories. The data is characterized by the imbalance of samples in different categories. The training set, validation set, and test set contain 9003, 1000, and 3080 samples respectively.

StackOverflow (Xu et al., 2015) is a dataset about programming languages released by Kaggle.com. The dataset is subdivided into 20 categories and has 20000 samples. The number of samples in the training set, validation set, and test set is 12000, 2000, and 6000 respectively. The detailed statistics of datasets are listed in Table 1.

4.2 Baselines

We compare with the following mainstream OOD intent classification methods: Softmax, MSP (Hendrycks and Gimpel, 2017), DOC (Shu et al., 2017), SEG (Yan et al., 2020), LMCL (Lin and Xu, 2019), OpenMax (Bendale and Boult, 2016), ADB (Zhang et al., 2021b), SCL (Zeng et al., 2021), KNN-CL (Zhou et al., 2022). The above baselines are described in detail in Related Work (Section 2).

For a fair comparison, BERT is the backbone network of all methods (The other backbone is also explored in Appendix C). We supplement the results of ADB (Zhang et al., 2021b), SCL(GDA+LOF) and KNN-CL (Zhou et al., 2022) based on their released codes, which are considered the state-of-the-art OOD classification methods at present, on three datasets. The other baselines adopt the results reported in respective papers.

4.3 Evaluation Metrics

We follow the metric adopted in previous work (Zhang et al., 2021b; Zeng et al., 2021; Zhou et al., 2022) and regard all OOD classes as a single rejected class. Accuracy and F1-score are calculated in the same way as Zeng et al. (2021); Zhou et al. (2022). **F1-IND** and **F1-OOD** represent macro F1-score values of IND and OOD classes respectively. To measure the overall effectiveness of the model, we also calculate the accuracy score and macro F1-score on all classes (including all IND and OOD classes) and denote them as **ACC-ALL** and **F1-ALL** respectively.

For the efficiency savings of the inference, we follow the settings in the research literature about early exit (Xin et al., 2020; Zhou et al., 2020; Liu et al., 2020; Sun et al., 2021). The specific calculation is as follows:

$$\text{Speedup} = \frac{\sum_{l=1}^{L} L \times N_l}{\sum_{l=1}^{L} l \times N_l}, \quad (12)$$

where L represents the number of layers, N_l is the number of samples which exit at layer l.

4.4 Experimental Setting

Similarly, following the previous work (Zhang et al., 2021b; Zhou et al., 2022), after splitting the dataset into the training set, validation set, and test set, we randomly retain 25%, 75% of the whole intent classes as IND classes and the remaining as OOD class. At the same, the samples with OOD class are excluded from the training set and preserved in the test set. All datasets undergo in the same processing to ensure the absence of OOD samples in the training set. For each configuration, we conduct at least three rounds of experiments with different random seeds and report the final average value.

We employ the widely adopted BERT model (bert-uncased, 12-layer transformer) provided by the Huggingface Transformers[2]. We fine-tune the model with the most commonly recommended hyperparameters. We adopt AdamW optimizer (Loshchilov and Hutter, 2019) and try learning rate in {1e-5, 2e-5, 5e-5, 5e-6}, the training

[2]https://github.com/huggingface/transformers

Dataset	Classes	|Training|	|Validation|	|Test|	Vocabulary	Length (Avg.)
CLINC-FULL (Larson et al., 2019)	150	15100	3100	5500	8288	8.32
BANKING (Casanueva et al., 2020)	77	9003	1000	3080	5028	11.91
StackOverflow (Xu et al., 2015)	20	12000	2000	6000	17182	9.18

Table 1: Statistics of CLINC-FULL, BANKING and StackOverflow datasets. || denotes the total number of utterances. Length indicates the average length of each utterance in the dataset.

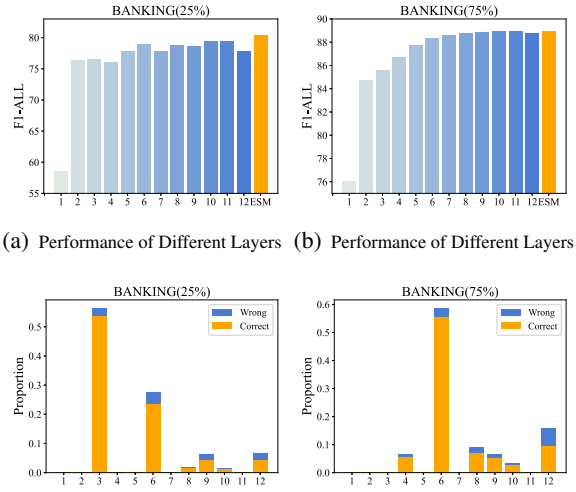
batch size in {16, 32}, the epochs 30 or 50. We fix the α and $\tilde{\alpha}$ as 1.0, and fix the β and $\tilde{\beta}$ as 0.1. We choose LOF (density-based and independent of the assumption of any distribution) as our detection method. According to Zhou et al. (2022), there is no distance to calculate **LOF** (Breunig et al., 2000) that is more advantageous than others. In this paper, we choose Cosine distance to calculate LOF. We use the same method as suggested in Zhou et al. (2022) to select the threshold of LOF by validation set. In addition, we also conduct experiments on other detection methods, such as KNN (Sun et al., 2022), to verify the generality and effectiveness of our method. All experiments are conducted on the NVIDIA GeForce RTX 3090 Graphical Card with 24G graphical memory.

4.5 Main Results

Table 2 shows the comparison between our method and baselines on BANKING and StackOverflow datasets, while the comparison between our method and baselines on CLINC-FULL is shown in Table 3. We highlight the best of all methods in bold. On the whole, our method can effectively improve the speed of inference on the premise of ensuring the accuracy of IND and OOD recognition.

Firstly, we observe the results in Table 2. We notice that our method performs particularly well on the BANKING dataset. With a speed-up ratio of more than 2 times, our method can not only ensure the accuracy of IND recognition but also effectively detect OOD, which fully demonstrates our method is both fast and good. The same phenomenon occurs in the comparison of the Stackoverflow and CLINC-FULL (Table 3) datasets. Under different settings, we exceed the existing baselines in all indicators. At the same time, We also maintain excellent inference efficiency.

On closer observation, we find that when the acceleration significantly increases, the effect also significantly improves, such as in BANKING and Stackoverflow(75%) datasets, and when the acceleration slows down, the effect also slows down, such as in CLINC-FULL(25%). This seems to show that the later the decision of the model is made, the closer the effect will be to the last level. This also verifies the necessity of introducing dynamic decision-making. The model itself can make decisions independently without needing to rely on the last layer to make decisions. We also conduct experiments with other backbone networks, see Appendix C for results.



(a) Performance of Different Layers (b) Performance of Different Layers

(c) Distribution of Exiting Samples (d) Distribution of Exiting Samples

Figure 3: Plots showing (**Top**) Performance of Different Layers and (**Bottom**) Distribution of Exiting Samples in (BANKING-25% (left) and BANKING-75%(right)). The **Top** shows the comparison of F1-ALL between each layer of the model and our method. The **Bottom** shows the distribution of exiting samples over the layers.

5 Analysis

5.1 A Closer Look at Internal Layers

Performance of Different Layers To better and more effectively show the effect of our method in a more fine-grained way, we have shown the performance of each layer of the model and the result of our method as shown in Figure 3(a)(b). First of all, we observe a more general form of "overthinking". After an internal layer of the model reaches the best performance, the performance begins to fluctuate, or even begins to decline(Figure 3(a)). At the

	Methods	BANKING					StackOverflow				
		ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
25%	MSP	43.67	50.09	41.43	50.55	1.00×	28.67	37.85	13.03	42.82	1.00×
	DOC	56.99	58.03	61.42	57.85	1.00×	42.74	47.73	41.25	49.02	1.00×
	OpenMax	49.94	54.14	51.32	54.28	1.00×	40.28	45.98	36.41	47.89	1.00×
	Softmax	57.88	58.32	62.52	58.10	1.00×	46.17	50.78	42.52	51.83	1.00×
	LMCL	64.21	61.36	70.44	60.88	1.00×	47.84	52.05	49.29	52.60	1.00×
	SEG	51.11	55.68	53.22	55.81	1.00×	47.00	52.83	46.17	54.16	1.00×
	SCL+GDA	83.87	67.94	89.44	66.81	1.00×	82.29	70.92	88.99	67.44	1.00×
	SCL+LOF	84.05	74.86	89.01	74.12	1.00×	80.10	78.51	84.45	77.32	1.00×
	ADB†	81.37	74.28	86.66	73.62	1.00×	86.69	79.24	90.93	76.91	1.00×
	KNN-CL†	86.26	78.19	90.64	77.53	1.00×	92.46	86.96	95.03	85.35	1.00×
	Ours	**89.00**$_{0.98}$	**80.35**$_{1.51}$	**92.70**$_{0.66}$	**79.70**$_{1.56}$	**2.41**×	**93.09**$_{0.64}$	**87.04**$_{1.05}$	**95.52**$_{0.43}$	**85.35**$_{1.19}$	**1.40**×
75%	MSP	75.89	83.60	39.23	84.36	1.00×	72.17	77.95	33.96	80.88	1.00×
	DOC	76.77	83.34	50.60	83.91	1.00×	68.91	75.06	16.76	78.95	1.00×
	OpenMax	77.45	84.07	50.85	84.64	1.00×	74.42	79.78	44.87	82.11	1.00×
	Softmax	78.20	84.31	56.90	84.78	1.00×	77.41	82.28	54.07	84.11	1.00×
	LMCL	78.52	84.31	58.54	84.75	1.00×	72.33	78.28	37.59	81.00	1.00×
	SEG	78.87	85.66	54.43	86.20	1.00×	80.83	84.78	62.30	86.28	1.00×
	SCL+GDA	79.86	85.14	64.49	85.5	1.00×	80.88	84.79	68.83	85.86	1.00×
	SCL+LOF	81.56	86.97	65.05	87.35	1.00×	80.92	83.98	71.71	84.79	1.00×
	ADB†	79.61	84.66	64.69	85.01	1.00×	81.11	84.40	72.26	85.21	1.00×
	KNN-CL†	82.69	87.18	71.17	87.46	1.00×	83.77	86.75	74.65	87.56	1.00×
	Ours	**85.20**$_{0.88}$	**88.98**$_{0.38}$	**74.94**$_{2.16}$	**89.23**$_{0.35}$	**1.64**×	**84.76**$_{1.96}$	**87.61**$_{1.14}$	**75.58**$_{4.57}$	**88.40**$_{0.91}$	**1.71**×

Table 2: OOD classification outcomes at 25% and 75% IND class rates on BANKING and StackOverflow. We attain the results of ADB (Zhang et al., 2021b) and KNN-CL (Zhou et al., 2022) through running their released codes. Other baselines are retrieved from Zhou et al. (2022). All reported results are percentages (except SPEEDUP) and the average over different seeds. The subscripts refer to corresponding standard deviations.

Methods	CLINC-FULL(25%)					CLINC-FULL(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
MSP	66.60	51.20	73.20	50.62	1.00×	73.41	81.81	57.83	82.02	1.00×
DOC	64.43	44.60	71.08	43.91	1.00×	74.63	78.63	64.62	78.76	1.00×
OpenMax†	68.50	61.99	75.76	61.62	1.00×	76.80	73.16	76.35	73.13	1.00×
Softmax	76.50	67.74	83.04	67.34	1.00×	86.26	89.01	83.12	89.61	1.00×
LMCL	68.57	62.42	75.61	62.01	1.00×	84.59	88.21	80.42	88.28	1.00×
SEG	72.86	65.44	79.90	65.06	1.00×	81.92	86.57	76.12	86.67	1.00×
ADB†	87.59	77.19	91.84	76.80	1.00×	86.32	88.53	83.92	88.58	1.00×
SCL+GDA‡	82.82	66.26	60.95	66.41	1.00×	83.14	84.93	84.82	84.94	1.00×
SCL+LOF‡	86.77	74.42	76.44	74.37	1.00×	81.55	84.04	78.57	84.09	1.00×
KNN-CL‡	92.11	82.04	95.07	81.69	1.00×	88.18	90.11	86.00	90.15	1.00×
Ours	**92.38**$_{0.26}$	**82.82**$_{0.28}$	**95.24**$_{0.19}$	**82.48**$_{0.29}$	**1.12**×	**89.32**$_{0.83}$	**91.03**$_{0.46}$	**87.41**$_{1.18}$	**91.06**$_{0.47}$	**1.67**×

Table 3: OOD classification results at 25% and 75% IND classes rates on CLINC-FULL. ‡ signifies the results are not furnished by SCL and KNN-CL, and we procure them through running their disclosed codes. The baselines with † are retrieved from (Zhang et al., 2021b), and the remaining baselines are derived from Zhan et al. (2021). All reported results are percentages (except SPEEDUP) and the average over different seeds different seeds. The subscripts refer to corresponding standard deviations.

same time, the early exit mechanism introduced in our method (orange) can effectively alleviate "overthinking" and ensure accuracy.

Distribution of Exiting Samples Figure 3(c)(d) further shows the distribution of exiting samples in detail. We can observe that most samples can exit in advance and maintain high accuracy. Figure 3(c)(d) further verify the effectiveness of our method, which can not only make samples exit early but also ensure the accuracy of intent recognition.

5.2 Importance of Training Methods

In this section, we explore the effectiveness of our training strategy. We compare different training methods with our method on the premise of keeping the inference strategy unchanged. **Average (AVG.)** refers to directly adding the losses of the internal classifiers as the training objective. **Joint** follows the training method in Zhou et al. (2020) and adds a weight coefficient to the loss of each layer. **Ensemble** is proposed by Sun et al. (2021) and divides the training loss into two parts, taking into account not only the relevance but also the diversity between different layers. See Appendix A

%	Methods	Banking		Stack.		Clinc	
		ACC.	F1	ACC.	F1	ACC.	F1
25	Avg.	87.18	77.83	91.78	85.30	89.84	78.89
	Joint	87.72	78.37	89.82	83.59	89.30	78.45
	Ensemble	88.14	78.89	90.32	82.68	90.64	79.80
	Ours	**89.00**	**80.35**	**93.09**	**87.04**	**92.38**	**82.82**
75	Avg.	83.37	87.31	82.86	86.09	85.75	88.00
	Joint	82.97	87.20	82.71	86.21	85.93	88.20
	Ensemble	83.21	87.35	82.97	85.94	87.55	89.62
	Ours	**85.20**	**88.98**	**84.76**	**87.61**	**89.32**	**91.03**

Table 4: Comparison results with different training methods. F1 is the F1-score for all classes. Detailed performance and discussion are available in Appendix A. All reported results are the average over different seeds.

%	Methods	Banking		Stack.		Clinc	
		ACC.	Speed	ACC.	Speed	ACC.	Speed
25	Random	84.93	1.83×	83.10	1.84×	87.18	1.84×
	Concat.	87.06	1.00×	91.88	1.00×	91.29	1.00×
	Pabee	87.47	2.04×	92.47	1.27×	91.37	1.59×
	Ours	**89.00**	2.41×	**93.09**	1.40×	**92.38**	1.12×
75	Random	82.28	1.83×	80.67	1.84×	86.39	1.84×
	Concat.	82.94	1.00×	83.33	1.00×	87.87	1.00×
	Pabee	84.58	1.90×	84.19	1.70×	88.51	2.14×
	Ours	**85.20**	1.64×	**84.76**	1.71×	**89.32**	1.67×

Table 5: Comparison results with different inference strategies on different datasets. ACC. is the accuracy for all classes. Detailed performance and discussion are available in Appendix B. All reported results are the average over different seeds.

for specific expressions of different training objectives. Our training methods are shown in Section 3.1. The comparison results are presented in Table 4. It can be seen from the table that our method can better adapt to the strategy adopted in inference. The comparison with **Ensemble** also shows the importance of reducing open space risk (Section 3.1) in our method. Detailed performance and discussion are the available in Appendix A.

5.3 Effect of Inference Strategy

In this section, we try to shed light on the effectiveness of our dynamic ensemble inference strategy. Under the same training strategy, we compare our strategy with other different inference strategies. **Random** means to the sample randomly early exits from a layer (the selection of layers follows uniform distribution). **Concat.** refers to concatenating the output representations of the internal layers, and takes the concatenated representation as the final representation. This approach is based on the previous research (Clark et al., 2019) that different layers of the model can capture the semantics of different levels of samples, and the fusion of different semantic representations may lead to better representation. **Pabee** is a widely used inference strategy proposed by Zhou et al. (2020). Refer to Appendix B for more details on strategies. The comparison results are presented in Table 5, from which it can be concluded that our inference strategy can achieve better results on each dataset. Detailed performance and discussion are available in Appendix B.

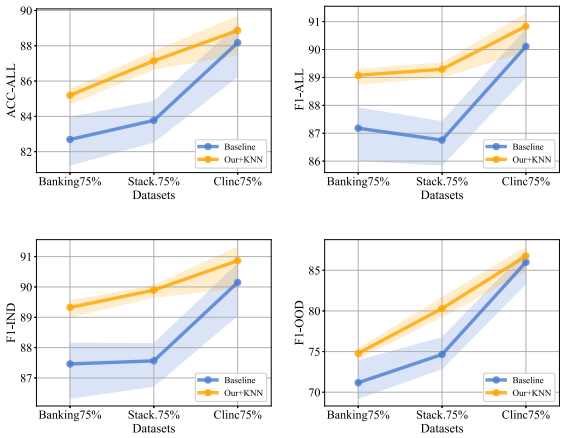


Figure 4: Effect of combining our method and KNN (Orange). The baseline is the result of KNN-CL (Blue) (Zhou et al., 2022).

5.4 Compatible with Other Detection methods?

In the above experiments and analysis, we have been using **LOF** (Breunig et al., 2000) as the OOD detection method (also known as a scoring function) and proved the effectiveness of our method based on it. To verify the generality of our proposed method proposed, i.e., whether it is compatible with other detection methods, we try another widely used distance-based OOD detection method–KNN (see (Sun et al., 2022) for details). After replacing it with KNN, we compare it with KNN-CL. As shown in Figure 4, our method also achieve better results in different datasets, which further proves the generality of our method.

6 Conclusion

In this paper, we explore whether the model would be overthinking in the open-world scenario and

demonstrate how to affect the discrimination between IND and OOD. On this basis, we propose a two-birds-one-stone method, i.e., during inference, let the model independently choose whether to exit early without going to the last layer, which not only ensures the accuracy of recognition but also accelerates the speed of inference. At the same time, we also introduce a training method that can adapt to the dynamic inference of the model. Detailed experiments and analysis show that our method can not only accelerate inference but also establish substantial improvements.

Limitations

To further inspire the follow-up work, we summarize our limitations as follows: 1) We only preliminarily reveal the overthinking phenomenon in the open-world scenario, and explore how to mitigate and utilize it during inference. We do not continue to conduct more in-depth research on the broader forms of overthinking in the open-world scenario and do not explore whether there are differences in its performance in different models. In addition, whether it can be solved or alleviated by other ways, such as training methods. 2) From the results of Sections 5.2, 5.3 and the corresponding Appendix A, B, it seems that there is room for further improvement in the speedup of our method. We leave how to achieve the best accuracy-speed trade-offs to subsequent research. 3) We have preliminarily verified that our method can be compatible with more detection algorithms and models, and look forward to exploring more methods and models.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No.2022CSJGG0801) and National Natural Science Foundation of China (No.62022027).

References

Abhijit Bendale and Terrance E Boult. 2016. Towards open set deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1563–1572.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

Inigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. *arXiv preprint arXiv:2003.04807*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Robert M Fano. 1961. Transmission of information: A statistical theory of communications. *American Journal of Physics*, 29(11):793–794.

Martin E. Hellman and Josef Raviv. 1970. Probability of error, equivocation, and the chernoff bound. *IEEE Trans. Inf. Theory*, 16(4):368–372.

Dan Hendrycks and Kevin Gimpel. 2017. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. 2019. Shallow-deep networks: Understanding and mitigating network overthinking. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 3301–3310. PMLR.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. 2019. An evaluation dataset for intent classification and out-of-scope prediction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages

1311–1316. Association for Computational Linguistics.

Ting-En Lin and Hua Xu. 2019. Deep unknown intent detection with margin loss. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5491–5496, Florence, Italy. Association for Computational Linguistics.

Ziqian Lin, Sreya Dutta Roy, and Yixuan Li. 2021. Mood: Multi-level out-of-distribution detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. 2020. FastBERT: a self-distilling BERT with adaptive inference time. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6035–6044, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. 2012. Toward open set recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1757–1772.

Lei Shu, Hu Xu, and Bing Liu. 2017. DOC: deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 2911–2916. Association for Computational Linguistics.

Tianxiang Sun, Yunhua Zhou, Xiangyang Liu, Xinyu Zhang, Hao Jiang, Zhao Cao, Xuanjing Huang, and Xipeng Qiu. 2021. Early exiting with ensemble internal classifiers. *arXiv preprint arXiv:2105.13792*.

Yiyou Sun, Yifei Ming, Xiaojin Zhu, and Yixuan Li. 2022. Out-of-distribution detection with deep nearest neighbors. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 20827–20840. PMLR.

Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. 2021. Open-set recognition: A good closed-set classifier is all you need. *CoRR*, abs/2110.06207.

Ji Xin, Raphael Tang, Jaejun Lee, Yaoliang Yu, and Jimmy Lin. 2020. Deebert: Dynamic early exiting for accelerating BERT inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2246–2251. Association for Computational Linguistics.

Jiaming Xu, Peng Wang, Guanhua Tian, Bo Xu, Jun Zhao, Fangyuan Wang, and Hongwei Hao. 2015. Short text clustering via convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 62–69.

Keyang Xu, Tongzheng Ren, Shikun Zhang, Yihao Feng, and Caiming Xiong. 2021. Unsupervised out-of-domain detection via pre-trained transformers. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1052–1061, Online. Association for Computational Linguistics.

Guangfeng Yan, Lu Fan, Qimai Li, Han Liu, Xiaotong Zhang, Xiao-Ming Wu, and Albert YS Lam. 2020. Unknown intent detection using gaussian mixture model with an application to zero-shot intent classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1050–1060.

Zhiyuan Zeng, Keqing He, Yuanmeng Yan, Zijun Liu, Yanan Wu, Hong Xu, Huixing Jiang, and Weiran Xu. 2021. Modeling discriminative representations for out-of-domain detection with supervised contrastive learning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 2: Short Papers), Virtual Event, August 1-6, 2021*, pages 870–878. Association for Computational Linguistics.

Li-Ming Zhan, Haowen Liang, Bo Liu, Lu Fan, Xiao-Ming Wu, and Albert Y. S. Lam. 2021. Out-of-scope intent detection with self-supervision and discriminative training. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3521–3532. Association for Computational Linguistics.

Hanlei Zhang, Xiaoteng Li, Hua Xu, Panpan Zhang, Kang Zhao, and Kai Gao. 2021a. TEXTOIR: An integrated and visualized platform for text open intent recognition. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 167–174, Online. Association for Computational Linguistics.

Hanlei Zhang, Hua Xu, and Ting-En Lin. 2021b. Deep open intent classification with adaptive decision boundary. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14374–14382.

Wangchunshu Zhou, Canwen Xu, Tao Ge, Julian J. McAuley, Ke Xu, and Furu Wei. 2020. BERT loses

patience: Fast and robust inference with early exit. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Yunhua Zhou, Peiju Liu, and Xipeng Qiu. 2022. Knn-contrastive learning for out-of-domain intent classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5129–5141. Association for Computational Linguistics.

Zhi-Hua Zhou and Nan Li. 2010. Multi-information ensemble diversity. In *Multiple Classifier Systems, 9th International Workshop, MCS 2010, Cairo, Egypt, April 7-9, 2010. Proceedings*, volume 5997 of *Lecture Notes in Computer Science*, pages 134–144. Springer.

Chen Zhu, Yu Cheng, Zhe Gan, Siqi Sun, Tom Goldstein, and Jingjing Liu. 2020. Freelb: Enhanced adversarial training for natural language understanding. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Method	**Loss**
Avg.	$\sum_{i=1}^{L} \mathcal{L}_{ce}^{i}$
Joint	$\frac{\sum_{i=1}^{L} i \cdot \mathcal{L}_{ce}^{i}}{\sum_{i=1}^{L} i}$
Ensemble	$\sum_{i=1}^{L} \mathcal{L}_{ce}^{i} - \lambda \sum_{i=2}^{L} \min_{j<i} \mathcal{L}_{ce}^{ij}$

Table 6: Comparison of different training methods. See Section 3.1 for our training method.

A More Results on the Comparison of Training Methods

In this section, we will show more detailed results of the comparison of different training strategies. First, we list the detailed training objectives in Table 6. Then, In Table 7, Table 8 and Table 9, we show the detailed comparison results of our methods and different training strategies on CLINC, BANKING and StackOverflow datasets. From the above comparison, we can find that our training strategy is not only effective but also general, which achieves better results in all datasets. From the comparison, our method seems to have room for improvement in the acceleration of inference, and we leave how to achieve better accuracy-speed trade-offs to subsequent research.

B More Results on the Comparison of Inference Strategy

In this section, we will show more detailed results of the comparison of different inference strategies. First, Let us review these strategies again. **Random** means to the sample randomly early exits from a layer, i.e., randomly select a layer of BERT according to the uniform distribution, and then the sample early exits from this layer. **Concat.** refers to concatenating the output representations of the internal layers and takes the concatenated representation as the final representation. This approach is based on the previous research (Clark et al., 2019) that different layers of the model can capture the semantics of different levels of samples, and the fusion of different semantic representations may lead to better representation. **Pabee** is a widely used inference strategy proposed by Zhou et al. (2020) and makes a decision according to whether the predictions of the just passed continuous k (called patience) classifiers (not all classifiers) are consistent, see Zhou et al. (2020) for details. Then, In Table 10, Table 11 and Table 12, we show the detailed comparison results of our methods and different inference strategies on BANKING, CLINC, and StackOverflow datasets.

From the above comparison in tables, we can find that our inference strategy can achieve better results in all datasets. The speed-up ratio of **Random** is fixed around the $1.84\times$ because the average number of exit layers of each sample is fixed equal to 6.5 in **Random** strategy so the speed-up ratio is about $12/6.5 \approx 1.846$. **Concat.** requires the output representation of all layers of the model, so it does not speed up ($1.00\times$). At the same time, we find that the **Pabee** strategy can also achieve good results (compared with KNN-CL in the Table 2 and Table 3), which also verifies the generality of the overtaking phenomenon and the rationality of making a decision early, which is in line with our expectations. We also observe that our method seems to have room for the acceleration of inference, and we leave how to achieve the best accuracy-speed trade-offs to subsequent research.

C Compatible with Other Backbone?

In Section 5.4, we have verified that our method could be compatible with other detection methods. In this section, we explore whether our method can adapt to other Backbones. Taking ALBERT (Lan et al., 2020) as an archetype, we compare our

Methods	CLINC-FULL(25%)					CLINC-FULL(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Avg.	89.84	78.89	93.57	78.49	1.62×	85.75	88.00	83.29	88.04	3.18×
Joint	89.30	78.45	93.19	78.05	1.31×	85.93	88.20	83.52	88.24	1.38×
Ensemble	90.64	79.80	94.13	79.42	1.81×	87.55	89.62	85.28	89.66	1.64×
Ours	**92.38**	**82.82**	**95.24**	**82.48**	1.12×	**89.32**	**91.03**	**87.41**	**91.06**	1.67×

Table 7: Detailed comparison results with different training methods in CLINC dataset.

Methods	BANKING(25%)					BANKING(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Avg.	87.18	77.83	91.42	77.12	2.37×	83.37	87.31	72.61	87.57	2.79×
Joint	87.72	78.37	91.82	77.66	2.03×	82.97	87.19	71.61	87.47	2.48×
Ensemble	88.14	78.89	92.11	78.19	2.50×	83.21	87.35	71.75	87.63	2.79×
Ours	**89.00**	**80.35**	**92.70**	**79.70**	2.41×	**85.20**	**88.98**	**74.94**	**89.23**	1.64×

Table 8: Detailed comparison results with different training methods in BANKING dataset.

Methods	StackOverflow(25%)					StackOverflow(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Avg.	91.78	85.30	94.59	83.44	3.00×	82.86	86.09	72.93	86.96	2.66×
Joint	89.82	83.59	93.16	81.68	2.36×	82.71	86.21	71.86	87.17	2.47×
Ensemble	90.31	82.68	93.64	80.49	2.24×	82.97	85.94	73.70	86.75	2.38×
Ours	**93.09**	**87.04**	**95.52**	**85.35**	1.40×	**84.76**	**87.61**	**75.58**	**88.40**	1.71×

Table 9: Detailed comparison results with different training methods in StackOverflow dataset.

Methods	BANKING(25%)					BANKING(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Random	84.93	75.21	89.84	74.44	1.83×	82.28	86.74	69.72	87.04	1.83×
Concat.	87.06	77.70	91.37	76.99	1.00×	82.94	87.00	72.05	87.26	1.00×
Pabee	87.47	78.94	91.60	78.27	2.04×	84.58	88.59	73.24	88.86	1.90×
Ours	**89.00**	**83.35**	**92.70**	**79.70**	2.41×	**85.20**	**88.98**	**74.94**	**89.23**	1.64×

Table 10: Detailed comparison results with different inference strategies on BANKING dataset.

Methods	Stackoverflow(25%)					Stackoverflow(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Random	83.10	75.99	88.37	73.51	1.84×	80.67	84.11	69.02	85.11	1.84×
Concat.	91.88	85.43	94.69	83.58	1.00×	83.33	86.05	74.89	86.79	1.00×
Pabee	92.47	86.49	95.07	84.77	1.27×	84.19	87.20	74.58	88.04	1.70×
Ours	**93.09**	**87.04**	**95.52**	**85.35**	1.40×	**84.76**	**87.61**	**75.58**	**88.40**	1.71×

Table 11: Detailed comparison results with different inference strategies on Stackoverflow dataset.

Methods	CLINC-FULL(25%)					CLINC-FULL(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
Random	87.18	71.78	92.11	71.23	1.84×	86.39	88.49	84.00	88.53	1.84×
Concat.	91.29	81.23	94.51	80.87	1.00×	87.87	89.79	85.73	89.83	1.00×
Pabee	91.37	79.64	94.71	79.23	1.59×	88.51	90.16	86.69	90.19	2.14×
Ours	**92.38**	**82.82**	**95.24**	**82.48**	1.12×	**89.32**	**91.03**	**87.41**	**91.06**	1.67×

Table 12: Detailed comparison results with different inference strategies on CLINC dataset.

method with the comparable baseline methods. The comparison results are shown in Table 13, which demonstrate that our method can also obtain improvements while accelerating inference. We establish preliminary exploration and left research on other models for future work.

Methods	BANKING(25%)					BNAKING(75%)				
	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP	ACC-ALL	F1-ALL	F1-OOD	F1-IND	SPEEDUP
MSP	42.25	50.40	39.72	50.97	1.00×	74.91	82.50	39.41	83.24	1.00×
DOC	76.80	70.46	82.50	69.83	1.00×	77.92	83.05	64.98	83.36	1.00×
OpenMax	76.91	65.42	83.41	64.47	1.00×	42.66	30.11	45.98	29.83	1.00×
Softmax(LOF)	80.30	72.93	85.81	72.25	1.00×	79.36	84.39	65.47	84.72	1.00×
LMCL	79.74	72.34	85.35	71.66	1.00×	79.27	83.85	66.49	84.15	1.00×
ADB	85.01	60.67	90.82	59.09	1.00×	53.77	55.44	51.00	55.52	1.00×
(K+1)-way	67.63	63.94	73.81	63.42	1.00×	80.49	85.66	64.41	86.02	1.00×
KNN-CL‡	84.50	75.42	89.41	74.68	1.00×	80.10	85.61	64.29	85.98	1.00×
Ours	**87.85**	**79.08**	**91.87**	**78.41**	**2.33**×	**81.63**	**86.70**	**67.09**	**87.04**	**2.51**×

Table 13: OOD classification outcome comparisons employing ALBERT. We procure the KNN-CL outcome through running the codes released in Zhou et al. (2022) (replace the backbone with ALBERT), and the results of the remaining baselines are attained by running the codes released in Zhang et al. (2021a) (replace the backbone with ALBERT). All reported results are percentages (except SPEEDUP) and the average over different seeds.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section "Limitations" (7th Section)

☑ A2. Did you discuss any potential risks of your work?
Section "Limitations" (7th Section)

☑ A3. Do the abstract and introduction summarize the paper's main claims?
"Abstract" and Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4.1

☑ B1. Did you cite the creators of artifacts you used?
Section 4.1

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
These datasets are available for all researchers in the NLP community.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
These datasets are only for scientific research and are available for all members of the NLP research community. We have adhered to the typical method of utilizing these resources.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
These datasets are only for scientific research and are available for all members of the NLP research community. We have adhered to the typical method of utilizing these resources.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4.1

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4.1

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4.4

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4.4

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4.4 and Section 4.5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4.4

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.