LREC-COLING 2024

Discriminative Language Model as Semantic Consistency Scorer for Prompt-based Few-Shot Text Classification

Zhipeng XIE, **Yahe LI**

School of Computer Science, Fudan University, China

xiezp@fudan.edu.cn

Outline

- **Background**
- Method
- Experimental Results
- Future Work

Pretraining-Finetuning Paradigm

Pretraining-Finetuning paradigm - de facto standard for NLU and NLG tasks

Pretrained Language Models:

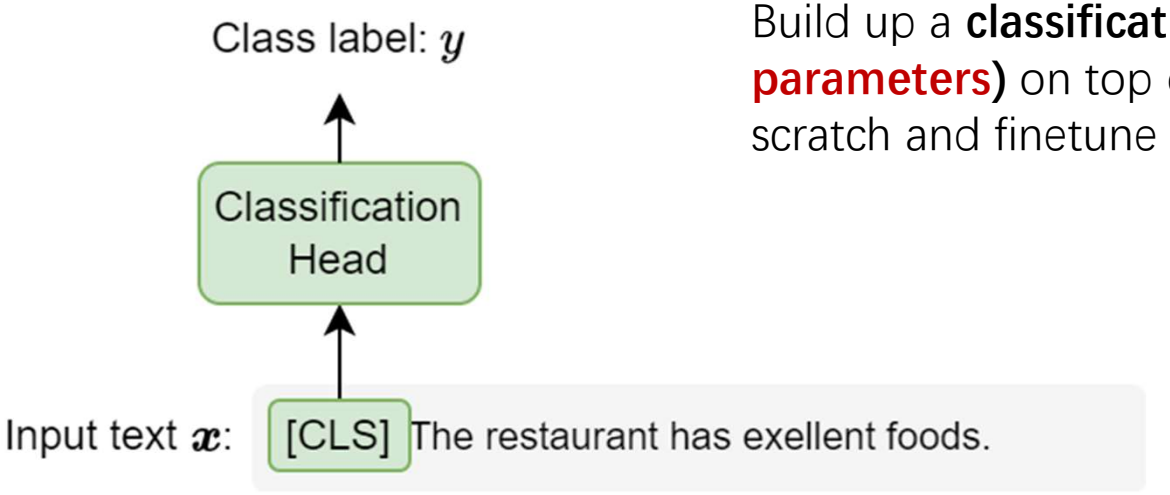
- **Autoregressive Models (ARM)** – Predict next token based on previous ones
  - ✓Effective for NLG tasks
- **Masked Language Model (MLM)** – Reconstruct masked tokens based on their bidirectional surrounding contexts
  - ✓Good at NLU tasks but usually not applicable to NLG

Finetuning PLM for Few Shot Text Classification

- **Few Shot Text Classification:** The amount of annotated examples is limited.

- **Two types of finetuning techniques:**
  - Conventional Finetuning
  - Prompt-based Finetuning
    - Masked Language Model
    - Discriminative Language Model

Conventional Finetuning



Build up a **classification head (with additional parameters)** on top of the special [CLS] token from scratch and finetune the whole model.

Conventional finetuning **works well with abundant training examples**, but will be **cornered in the few shot scenario**.

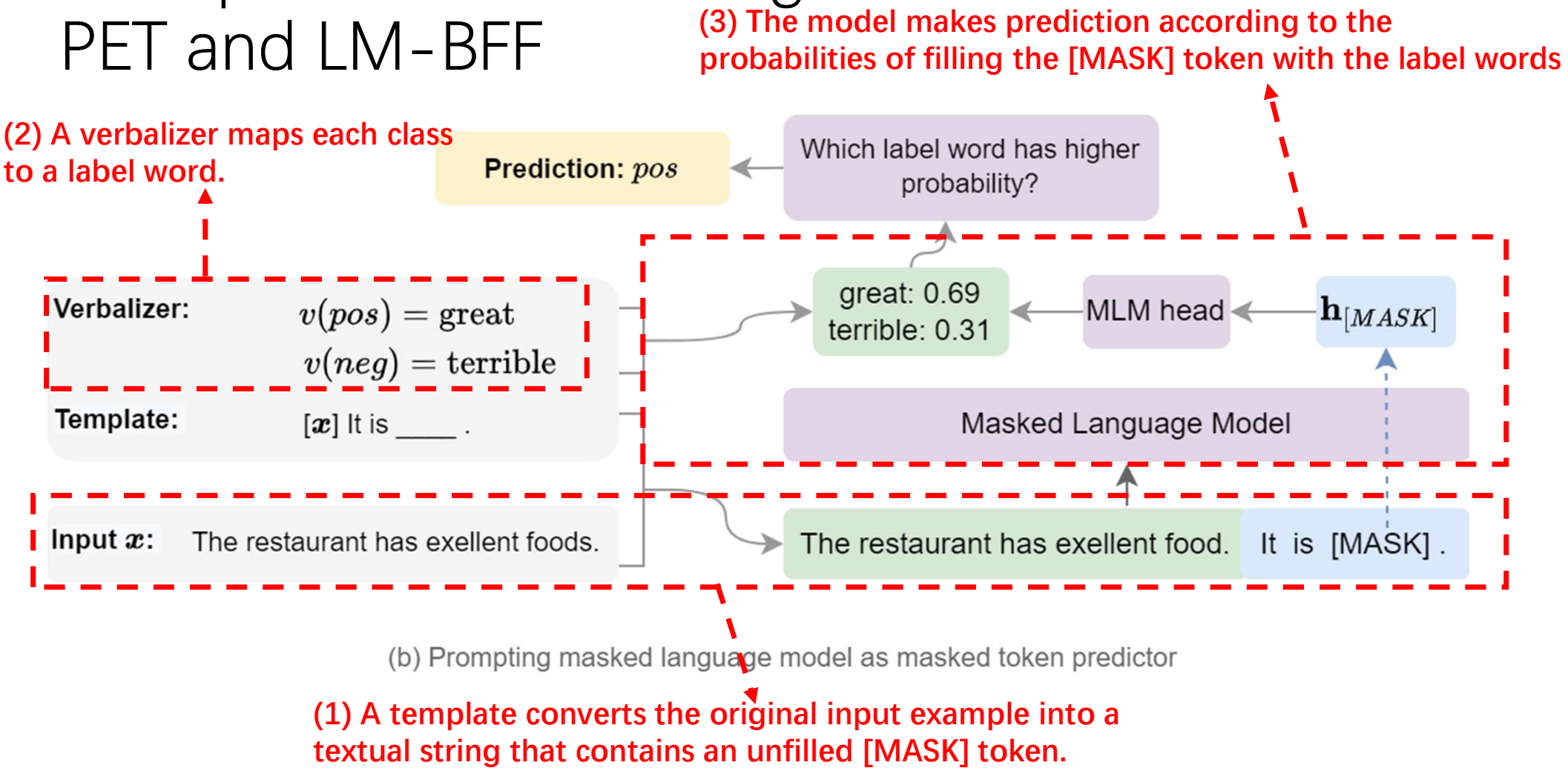
Prompt-based Finetuning

- **Prompt-based prediction** was originally developed by the GPT series for zero-shot predicitons
  - (Radford et al., 2018, 1029; Brown et al, 2020)

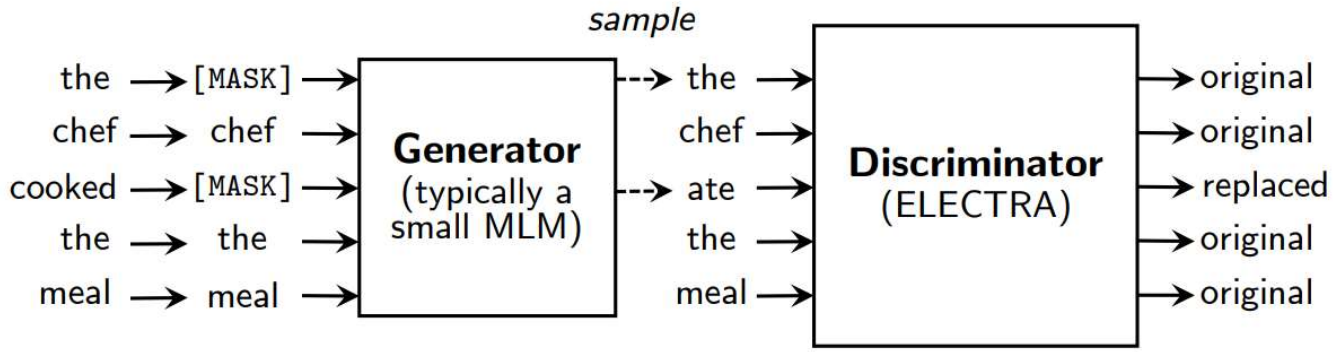
- **PET method** studied prompt-based prediction for **finetuning**
  - (Schick and Schutze, 2021a,b)

- **LM-BFF method** automated the process of **prompt generation**
  - (Gao et al., 2021)

Prompt-based Finetuning for Text Classification
PET and LM-BFF



ELECTRA – A Discriminative LM (Clark et al., 2020)



- The pretraining task: **replaced token prediction**.
  - The **generator** is trained to perform MLM task
  - The **discriminator** is trained to distinguish "real" tokens from "replaced" tokens
- The discriminator in **ELECTRA** consists of two modules:
  - An encoder: maps a sequence of input tokens $x = [x_1, \ldots, x_n]$into a sequence of contextualized vector representations $[\mathbf{h}_1, \ldots, \mathbf{h}_n]$
  - A discriminative head: predicts whether each token is a "real" or "replaced" token.

Prompting DLM for Text Classification
DPT (Yao et al., 2022)

- The DPT method converts each input text x into the following template of discriminative prompt:

$$\texttt{[CLS]}\ \boldsymbol{x}\ \texttt{Class:}\ v(l_1), v(l_2), \ldots, v(l_n).\ \texttt{[SEP]}$$

Input example: The restaurant has excellent food.

Discriminative Prompt: [CLS] The restaurant has excellent food. Class: great, terrible. [SEP]

The prompt in this style is **not natural** and is **not compatible** with the training data used for pretraining.

Prompting DLM for Text Classification PromptELECTRA (Xia et al., 2022)

- **PromptELECTRA** generates one discriminative prompt for each possible class label.

- DLM head is used to output the label word that has the highest probability of being original token in its corresponding prompt.

The prompts in this style is much more natural. But this method simply rely on the evidence from the label words, other possible evidences get ignored

Outline

- Introduction
- **Method**
- Experimental Results
- Future Work

Motivation

Input sentence: x = “The restaurant has excellent food.”

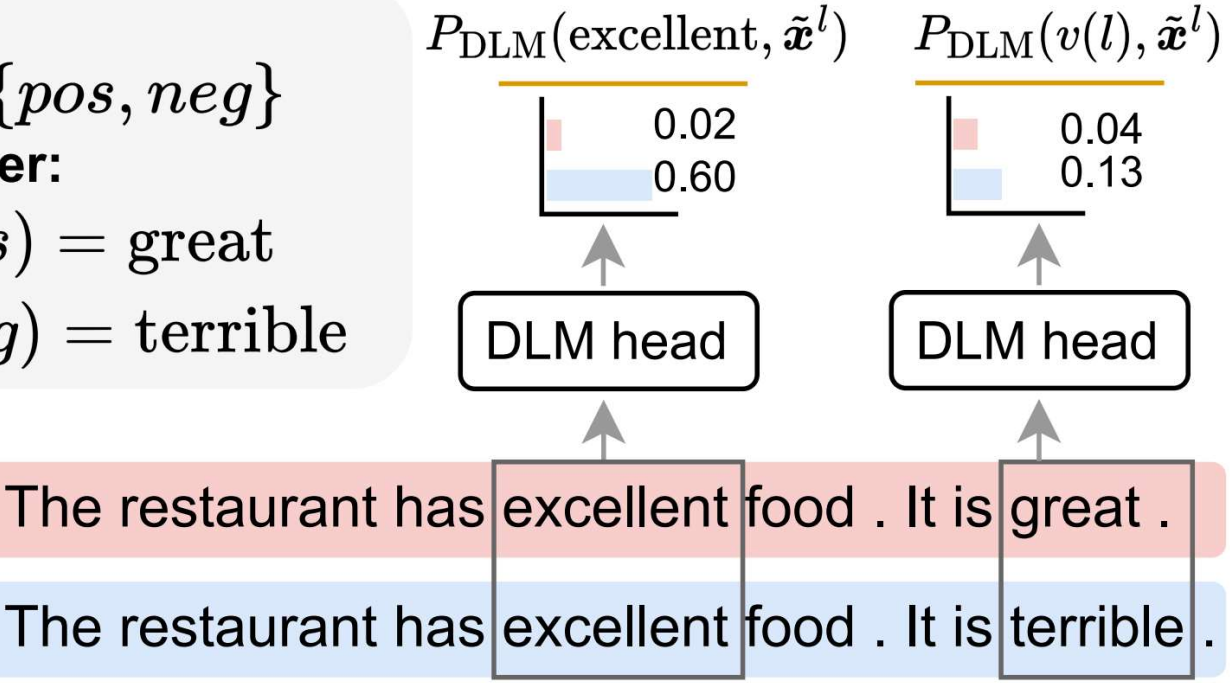
Lables:

$$\mathcal{L} = \{pos, neg\}$$

Verbalizer:

$$v(pos) = \text{great}$$

$$v(neg) = \text{terrible}$$



Method

- The **DLM-SCS** method stands for **Discriminative Language Model as Semantic Consistency Scorer**
- **Basic Idea: reformulate text classification as a task of semantic consistency scoring. The class label with the highest semantic consistency score is predicted:**

$$\hat{l} = \arg\max_{l \in \mathcal{L}} SC(\tilde{\boldsymbol{x}}^l)$$

- **Technique: How to calculate the semantic consistency of each discriminative prompt?**
  - **The weighted average of semantic consistency scores of multiple parts in the prompt.**

Discriminative Prompts
Sentence Classification

- Given an input example of sentence $\boldsymbol{x}_{\text{in}} = \boldsymbol{x}^{(1)}$

- The discriminative prompt for each label $l \in \mathcal{L}$ is generated as:

$$\tilde{\boldsymbol{x}}^{l} = [\text{CLS}]\ \boldsymbol{x}^{(1)}\ It\ is\ v(l)\ .\ [\text{SEP}]$$

- Each discriminative prompt $\tilde{\boldsymbol{x}}^{l}$ has two parts:
  - The sentence $\boldsymbol{x}^{(1)}$
  - The label word $v(l)$

Discriminative Prompts
Sentence Pair Classification

- Given an input example of sentence pair $\boldsymbol{x}_{\text{in}} = \left(\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\right)$

- The discriminative prompt for each label $l \in \mathcal{L}$ is generated as:

$$\tilde{\boldsymbol{x}}^{l} = [\text{CLS}]\ \boldsymbol{x}^{(1)}\ ?\ v(l), \boldsymbol{x}^{(2)}\ [\text{SEP}]$$

- Each discriminative prompt $\tilde{\boldsymbol{x}}^{l}$ has three parts:
  - The first sentence $\boldsymbol{x}^{(1)}$
  - The second sentence $\boldsymbol{x}^{(2)}$
  - The label word $v(l)$

Semantic Consistency Score: Discriminative Prompt

Semantic consistency score of a discriminative prompt

$$SC(\tilde{\boldsymbol{x}}^l) = \lambda_0 \cdot sc\left(v(l), \tilde{\boldsymbol{x}}^l\right) + \sum_{\boldsymbol{x}^{(i)} \in \boldsymbol{x}_{\text{in}}} \lambda_i \cdot sc\left(\boldsymbol{x}^{(i)}, \tilde{\boldsymbol{x}}^l\right)$$

Semantic consistency scores of multiple parts in the prompt

Semantic Consistency Score
Token Subsequence – Uniform Weights

Let s be a token subsequence in the discriminative prompt $\tilde{x}^l$, its semantic inconsistency is measured as:

$$sc(\boldsymbol{s}, \tilde{\boldsymbol{x}}^l) = \frac{\exp\left(-\frac{1}{|\boldsymbol{s}|}\sum_{x\in\boldsymbol{s}} \mathbf{w}^\top \mathbf{h}_x^l\right)}{\sum_{l'\in\mathcal{L}} \exp\left(-\frac{1}{|\boldsymbol{s}|}\sum_{x\in\boldsymbol{s}} \mathbf{w}^\top \mathbf{h}_x^{l'}\right)}$$

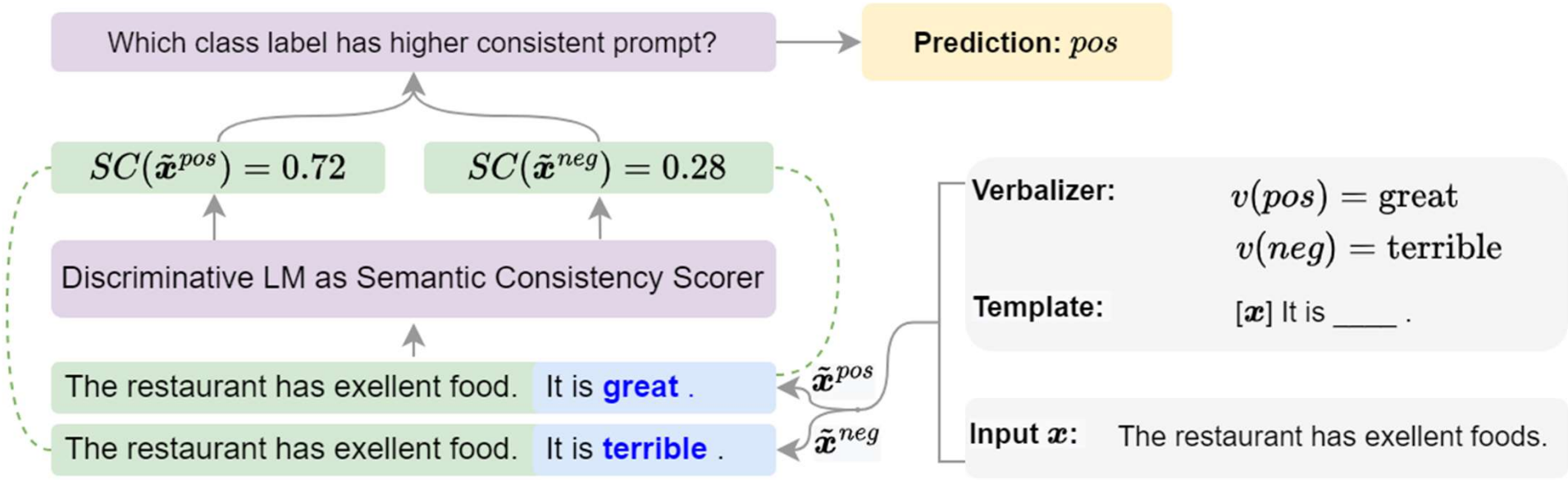
Semantic Consistency Score
Token Subsequence – IDF Weights

Different tokens should be of different importance with respect to the semenatic consistency

using IDF value as the weights:

$$sc(\boldsymbol{s}, \tilde{\boldsymbol{x}}^l) = \frac{\exp\left(-\frac{\sum_{x \in \boldsymbol{s}} \text{idf}(x) \mathbf{w}^\top \mathbf{h}_x^l}{\sum_{x \in \boldsymbol{s}} \text{idf}(x)}\right)}{\sum_{l' \in \mathcal{L}} \exp\left(-\frac{\sum_{x \in \boldsymbol{s}} \text{idf}(x) \mathbf{w}^\top \mathbf{h}_x^{l'}}{\sum_{x \in \boldsymbol{s}} \text{idf}(x)}\right)}$$

DLM-SCS: Schematic Illustration



(a) Prompting discriminative language model as semantic consistency scorer

Outline

- Background
- Method
- **Experimental Results**
- Future Work

Experimental Setup
Datasets

- Datasets:
  - 4 Sentence Classification Datasets: **SST-2**, **SST-5**, **MR** and **CR**
  - 6 Sentence-Pair Classification Datasets: **SNLI**, **MNLI**, **QNLI**, **RTE**, **MRPC** and **QQP**
- The number of training examples:
  - $K = 16$ (by default) training examples per class
  - The total number of training examples is $K \times |\mathcal{L}|$
  - Development set for model selection and hyperparameter tuning is of the same size as the training set

Experimental Setup
Templates and Label Words

- We adopt the templates and label words from (Gao et al., 2021):

Task	Template	Label words
SNLI	$<S_1>$? $v(l)$, $<S_2>$	Yes/No/Maybe
MNLI	$<S_1>$? $v(l)$, $<S_2>$	Yes/No/Maybe
QNLI	$<S_1>$? $v(l)$, $<S_2>$	Yes/No
RTE	$<S_1>$? $v(l)$, $<S_2>$	Yes/No
MRPC	$<S_1>$? $v(l)$, $<S_2>$	Yes/No
QQP	$<S_1>$. $v(l)$, $<S_2>$	Yes/No
SST-2	$<S_1>$It is $v(l)$.	terrible/great
SST-5	$<S_1>$It is $v(l)$.	terrible/bad/ okay/good/great
MR	$<S_1>$It is $v(l)$.	terrible/great
CR	$<S_1>$It is $v(l)$.	terrible/great

Experimental Setup
The Competitors

- **Fine-tuning:** the traditional fine-tuning of Roberta-Large
- **LM-BFF (man):** few-shot finetuning with manual prompts
- **LM-BFF (auto):** few-shot finetuning with automatically searched templates.
- Two prompt methods for discriminative language model:
  - **DPT**
  - **PromptELECTRA**

Main Results

Model	SNLI (acc)	MNLI (acc)	QNLI (acc)	RTE (acc)	MRPC (F1)	QQP (F1)	SST-2 (acc)	SST-5 (acc)	MR (acc)	CR (acc)
Fine-tuning	48.4 (4.8)	45.8 (6.4)	60.2 (6.5)	54.4 (3.9)	76.6 (2.5)	60.7 (4.3)	81.4 (3.8)	43.9 (2.0)	76.9 (5.9)	75.8 (3.2)
LM-BFF (man)	77.2 (3.7)	68.3 (2.3)	64.5 (4.2)	69.1 (3.6)	74.5 (5.3)	65.5 (5.3)	92.7 (0.9)	47.4 (2.5)	87.0 (1.2)	90.3 (1.0)
+demonstrations	79.7 (1.5)	70.7 (1.3)	69.2 (1.9)	68.7 (2.3)	77.8 (2.0)	69.8 (1.8)	92.6 (0.5)	50.6 (1.4)	86.6 (2.2)	90.2 (1.2)
LM-BFF (auto)	77.1 (2.1)	68.3 (2.5)	68.3 (7.4)	73.9 (2.2)	76.2 (2.3)	67.0 (3.0)	92.3 (1.0)	49.2 (1.6)	85.5 (2.8)	89.0 (1.4)
+demonstrations	77.5 (3.5)	70.0 (3.6)	68.5 (5.4)	71.1 (5.3)	78.1 (3.4)	67.7 (5.8)	93.0 (0.6)	49.5 (1.7)	87.7 (1.4)	91.0 (0.9)
DART	75.8 (1.6)	67.5 (2.6)	66.7 (3.7)	68.7 (1.3)	**78.3** (4.5)	67.8 (3.2)	93.5 (0.5)	49.6 (0.9)	88.2 (1.0)	**91.8** (0.5)
DPT	47.4 (7.7)	39.0 (1.8)	54.6 (5.4)	50.2 (2.8)	76.4 (6.1)	56.1 (1.1)	92.6 (1.3)	44.0 (3.8)	89.5 (2.1)	91.2 (1.6)
PromptELECTRA	79.1 (3.4)	65.8 (2.5)	70.9 (2.1)	68.2 (2.8)	73.5 (4.6)	63.1 (3.3)	93.1 (1.0)	51.4 (2.2)	89.4 (1.6)	90.2 (1.4)
DLM-SCS (ours)	**82.2** (1.5)	**71.0** (2.0)	**77.0** (2.4)	**75.0** (2.9)	**78.3** (3.1)	**72.2** (1.4)	**93.6** (0.6)	**51.5** (2.0)	**90.2** (0.7)	91.0 (1.4)

Mean performance of the runner-up: 72.7

Mean Performance of Our method: 76.0

Mean performance of the runner-up: 81.0

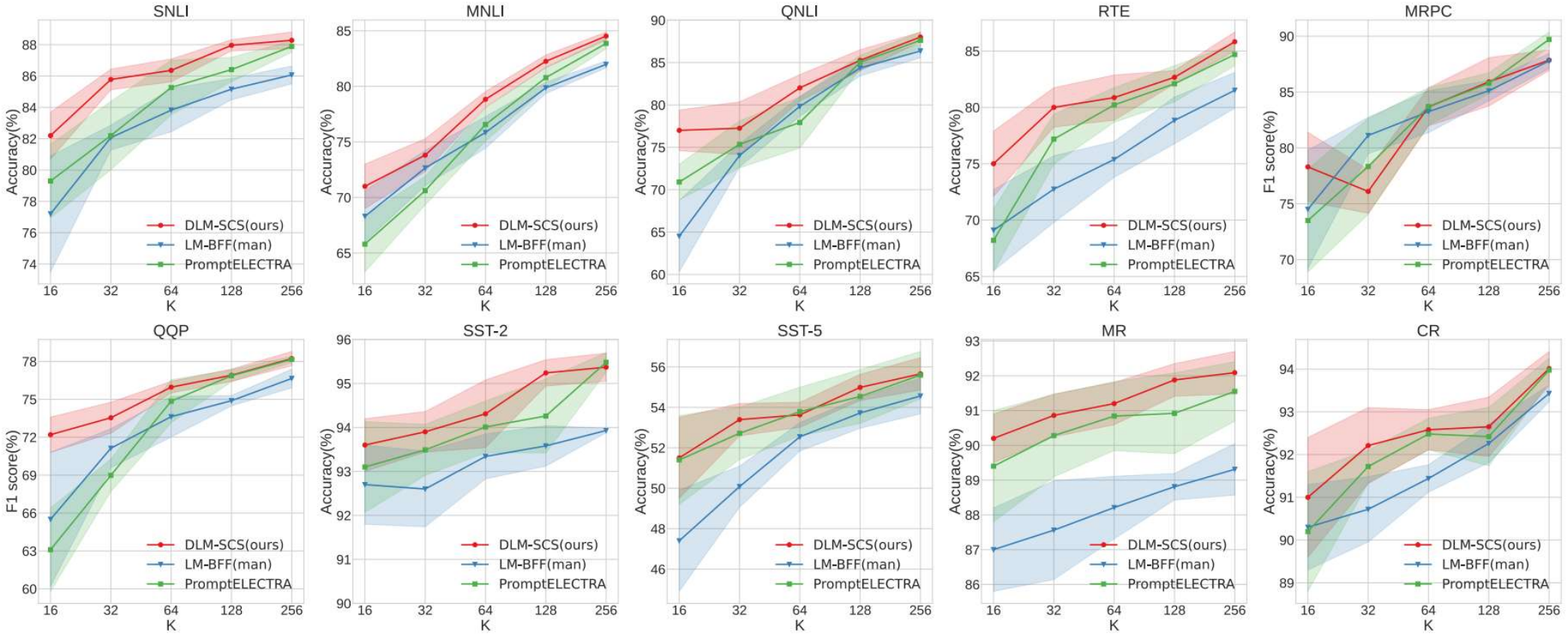
Mean Performance of Our method: 81.6

Ablation Analysis

Two main techniques in DLM-SCS:
1) **Integrating the evidences from multiple components** (or parts) of the prompt
2) **weighting the tokens** in each prompt part with IDF values

Model	SNLI	MNLI	QNLI	RTE	MRPC	QQP	SST-2	SST-5	MR	CR
(full) DLM-SCS	**82.2**	**71.0**	**77.0**	**75.0**	**78.3**	**72.2**	93.6	**51.5**	90.2	**91.0**
-w.o. token weight	78.2	70.0	73.4	73.6	76.9	69.6	93.0	48.8	**90.3**	90.3
-only label word	76.5	64.6	69.0	71.8	74.8	64.2	**93.7**	51.1	88.8	90.4

Varying the size of training examples



Outline

- Background
- Method
- Experimental Results
- **Future Work**

Two Future Directions

- How to integrate our discriminative framework with auotomatic prompt generation and differentiable prompting

- How to combine our discriminative framework with generative PLMs?

Thanks for Listening! Any Questions?