Few-Shot Anaphora Resolution in Scientific Protocols via Mixtures of In-Context Experts

Nghia T. Le, Fan Bai, Alan Ritter
School of Interactive Computing
Georgia Institute of Technology
{nle18,fan.bai,alan.ritter}@cc.gatech.edu

Abstract

Anaphora resolution is an important task, which traditionally has required costly supervised training datasets for each new language, text genre, and domain. Meanwhile, prompting large language models with a few in-context examples has emerged as a promising approach to reduce labeling costs, however there are a number of challenges in applying in-context learning to resolve anaphora. In this paper, we present MICE (Mixtures of In-Context Experts), which we demonstrate is effective for few-shot anaphora resolution in the domain of scientific protocols (Tamari et al., 2021). Given only a handful of training examples, MICE combines the predictions of hundreds of in-context experts, yielding a 30% increase in F_1 score over a competitive prompt retrieval baseline. Furthermore, we show MICE can be used to train compact student models without sacrificing performance. As far as we are aware, this is the first work to present experimental results demonstrating the effectiveness of in-context learning on the task of few-shot anaphora resolution in scientific protocols.[1]

1 Introduction

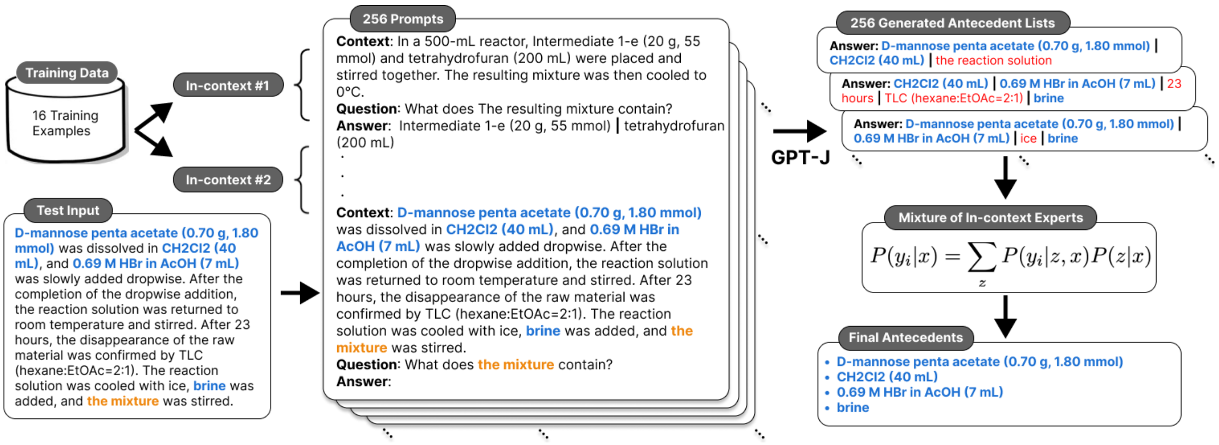
Prompting large language models (LMs) with in-context demonstrations has enabled surprisingly effective few-shot learning (Brown et al., 2020). However, more complex linguistic annotations over paragraph-length inputs, such as anaphora and coreference, have proven challenging (Yang et al., 2022). Prompting language models with demonstrations of anaphora and their corresponding antecedents requires encoding long sequences of tokens, limiting the number of demonstrations that can be used within a single prompt. Furthermore, the performance of in-context learning has been shown to be sensitive to the choice of demonstrations (Liu et al., 2022b) and their ordering in the prompt (Lu et al., 2022).

[1]Our code and datasets are available at `https://github.com/nle18/mice`

To address these challenges, this paper presents **M**ixtures of **I**n-**C**ontext **E**xperts (MICE). We demonstrate the effectiveness of MICE on anaphora resolution in chemical synthesis protocols (see examples in Figure 1). Scientific protocols make an ideal testbed for few-shot anaphora resolution because they contain rich coreference and bridging links. Furthermore, these linguistic structures are expressed very differently from well-studied domains with extensive annotated resources, such as newswire, and they are not easily amenable to annotation by non-expert crowd workers.

MICE works as follows. Given an anaphor, such as *"the mixture"*, it uses in-context learning to predict a list of substances contained in the mixture that are referenced earlier in the procedure, for example: *"Bromoacetyl bromide"*, *"compound 54"* and *"water"*. With only a handful of training examples (e.g., 16 or 32), MICE generates an ensemble of up to k^d in-context experts, each of which consists of a prompt containing d demonstrations chosen from the k available training examples. For instance, if $d = 2$ in-context demonstrations per prompt and $k = 16$ training examples, MICE generates up to 256 in-context experts (prompts) to be ensembled. The experts' predictions are then combined in a mixture model, where mixture weights are computed by comparing embeddings of the input mention and available training examples.

We find that, although some prompts and prompt orderings perform better than others, individual prompts act as local experts in different regions of the input space (Jacobs et al., 1991; Jordan and Jacobs, 1994; Shazeer et al., 2017), and no single prompt works better than others on all inputs (see Figure 2). Furthermore, if the same antecedent is predicted by multiple in-context experts, this provides independent sources of evidence, increasing the probability the answer is correct (Downey et al., 2005). In extensive experiments, we show MICE significantly improves the performance of



Test Input: D-mannose penta acetate (0.70 g, 1.80 mmol) was dissolved in CH2Cl2 (40 mL), and 0.69 M HBr in AcOH (7 mL) was slowly added dropwise. After the completion of the dropwise addition, the reaction solution was returned to room temperature and stirred. After 23 hours, the disappearance of the raw material was confirmed by TLC (hexane:EtOAc=2:1). The reaction solution was cooled with ice, brine was added, and the mixture was stirred.

Context: In a 500-mL reactor, Intermediate 1-e (20 g, 55 mmol) and tetrahydrofuran (200 mL) were placed and stirred together. The resulting mixture was then cooled to 0°C.
Question: What does The resulting mixture contain?
Answer: Intermediate 1-e (20 g, 55 mmol) | tetrahydrofuran (200 mL)
.
.
.
Context: D-mannose penta acetate (0.70 g, 1.80 mmol) was dissolved in CH2Cl2 (40 mL), and 0.69 M HBr in AcOH (7 mL) was slowly added dropwise. After the completion of the dropwise addition, the reaction solution was returned to room temperature and stirred. After 23 hours, the disappearance of the raw material was confirmed by TLC (hexane:EtOAc=2:1). The reaction solution was cooled with ice, brine was added, and the mixture was stirred.
Question: What does the mixture contain?
Answer:

Answer: D-mannose penta acetate (0.70 g, 1.80 mmol) | CH2Cl2 (40 mL) | the reaction solution
Answer: CH2Cl2 (40 mL) | 0.69 M HBr in AcOH (7 mL) | 23 hours | TLC (hexane:EtOAc=2:1) | brine
Answer: D-mannose penta acetate (0.70 g, 1.80 mmol) | 0.69 M HBr in AcOH (7 mL) | ice | brine

- D-mannose penta acetate (0.70 g, 1.80 mmol)
- CH2Cl2 (40 mL)
- 0.69 M HBr in AcOH (7 mL)
- brine

Figure 1: Resolving antecedents of *"the mixture"* in a chemical synthesis procedure using MICE. Given a small training set of 16 examples and a test input, we construct 256 prompts, each with two in-context demonstrations. The prompts are then fed into a pre-trained language model (e.g. GPT-J) to generate candidate antecedents. The probabilities of each candidate antecedent are computed and combined in a mixture of in-context experts using a similarity-based gating function. MICE then selects the antecedents with the highest probabilities. In the figure, orange, blue, red denote **the anaphor**, **the true antecedents**, and incorrect antecedents, respectively.

in-context learning for anaphora resolution in synthetic procedures. For example, given 32 demonstrations, using a single prompt achieves an F_1 score of 38.6, whereas by combining the predictions of 256 prompts, MICE achieves 53.9 F_1.

While MICE consistently improves in-context anaphora resolution, inference is relatively expensive, due to the large number of prompts involved. To address this limitation, we show that fine-tuning BERT-based models on data that is automatically labeled with MICE yields further performance improvements, while also producing much more compact models (Schick and Schütze, 2021a,b; Lang et al., 2022).

2 Anaphora Resolution in Scientific Protocols

Split-antecedent anaphors (Vala et al., 2016; Yu et al., 2020; Paun et al., 2022) are plural mentions that refer to two or more antecedents in the previous discourse. For instance in the following text: "`[Alice]`$_{\texttt{antecedent}}$ `and [Bob]`$_{\texttt{antecedent}}$ `went to the store. [They]`$_{\texttt{anaphor}}$ `bought some bread.`" the word "`[They]`" refers to two both "`[Alice]`" and "`[Bob]`".

Similar references to multiple antecedents appear in chemical synthesis protocols, for example, "*the mixture*". These references arise naturally as the result of context change accommodation (Webber and Baldwin, 1992), and are crucial for understanding the steps needed to synthesize a molecule (Fang et al., 2021). Resolving anaphoric references in synthetic protocols could be beneficial for automating protocols described in natural language (Sanderson, 2019; Vaucher et al., 2021), in addition to automatically extracting chemical reaction databases from scientific literature (Lawson et al., 2014; Mysore et al., 2019). However, anaphora is costly to annotate (Yuan et al., 2022) and scientific

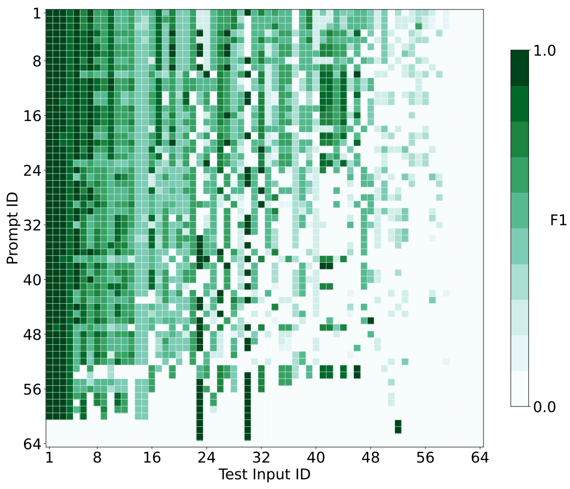


Figure 2: Heatmap visualizing the performance of 64 prompts on 64 sampled anaphors. Each prompt encodes two in-context demonstrations randomly sampled from 8 training examples. Each square represents F_1 of a single prompt applied to a single anaphor (typically these are associated with multiple antecedents). The prompts and test inputs are sorted from high (top, left) to low (bottom, right) F_1. Note that no single prompt performs best on all test inputs. This suggests that it could be beneficial to combine lists of predicted antecedents made independently by many *in-context experts*.

protocols are not easily amenable to annotation by non-expert crowd workers (Kulkarni et al., 2018). This motivates the need for few-shot learning methods that can resolve anaphora in procedural texts without extensive annotated resources.

3 Mixtures of In-Context Experts

While in-context learning has achieved good performance when prompted with a few examples, the performance can vary significantly depending on different prompt design choices (Lu et al., 2022; Liu et al., 2022b). Furthermore, anaphora resolution requires paragraph-length contexts, limiting the number of in-context examples that can be encoded in a single prompt (Figure 3). We address these challenges with MICE. We show MICE is an effective method for few-shot anaphora resolution in §5, and demonstrate that it can be used to automatically label data for fine-tuning more compact models, without sacrificing performance, in §4.1.

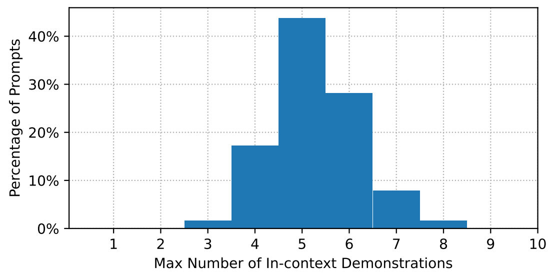


Figure 3: Distribution of the maximum number of in-context demonstrations of an anaphor, synthesis protocol, and corresponding antecedents that can be encoded in a single prompt. We compute the max number of demonstration tokens by subtracting the max sequence length (2048) by a fixed number for generated tokens (256) and the number of tokens for the test input. Demonstrations are randomly sampled until the max number of tokens is reached. Given the longer contexts needed to demonstrate anaphora resolution, a prompt can encode at most 8 demonstrations, much less than the 32 used in Brown et al. (2020).

In-Context Learning We formulate the task of anaphora resolution in synthetic protocols as follows. The input includes a document D and a query anaphor a. Our goal is to identify a set of antecedents $\mathcal{Y} = \{y_0, y_1, ..., y_m\}$ that correspond to text spans in D. To tackle this problem via in-context learning, we frame it as a SQuAD-style extractive question answering task (Wu et al., 2020). Specifically, as shown in Figure 1, each example (D, a) is formatted as the concatenation of document D and template question: "`What does` a `contain?`" An autoregressive language model then completes this sequence by generating $\mathcal{Y}$, with the antecedents separated by a special marker "`|`". Following the typical approach to in-context learning, the prompt includes a few demonstrations in the prefix and ends with the test input.

Mixture of Experts For a given test input $x = (D, a)$, we aim to find the antecedents y_i with the highest probabilities $P(y_i|x)$. The notation y_i denotes an antecedent from the union of antecedents generated by all prompts. MICE computes $P(y_i|x)$ using a mixture of experts (Jacobs et al., 1991; Cho et al., 2019), treating the prompt, z, as a latent variable (Guu et al., 2020):

$$P(y_i|x) = \sum_z P(y_i|z, x)P(z|x) \tag{1}$$

In Eq.1, $P(z|x)$ represents the likelihood that prompt z is constructed given x, and $P(y_i|z, x)$ represents the probability that the LM predicts antecedent y_i when prompted with z and x.

Similarity-based Gating We compute $P(z|x)$ by summing similarity scores $s(x, u_1), ..., s(x, u_d)$ between x and the in-context demonstrations $u_1, ..., u_d$ encoded in z:[2]

$$P(z|x) \propto \exp \sum_{i=1}^{d} s(x, u_i)$$

where $s(x, u_i)$ is the cosine similarity between the embeddings of x and u_i. Details of the similarity measures used in our experiments are presented in §4.1.

Estimating Antecedent Probabilities Computing probabilities $P(y_i|z, x)$ that are comparable across variable-length antecedents is not easy. Longer sequences will naturally have smaller LM probabilities, suggesting the need for length normalization, or averaging per-token probabilities, neither of which we found to work well.[3] Therefore, following Zhao et al. (2021), we estimate $P(y_i|z, x)$ using first token probabilities. Specifically, let $y_{i,0}$ denote the first token of y_i. We then

[2]We also experimented with multiplying the similarity scores and observed similar results.

[3]Similar to Zhao et al. (2021), we observe that, for a generated antecedent, the first token probabilities vary the most, while probabilities of subsequent tokens are highly deterministic.

have:

$$P(y_i|z,x) \approx \max_j P_j(y_{i,0}|z,x) \qquad (2)$$

where the max is taken over $P_j(y_{i,0}|z,x)$ – the probability of token $y_{i,0}$ being the first token of the jth antecedent generated by the language model, when prompted with z and x.

Approximation with Sampling Approximating $P(y_i|z,x)$ using first-token probabilities in Eq.2 has a drawback: it disregards the length of the antecedents and treats different antecedents with the same first token as equivalent. As an alternative, we present MICE-SAMPLING: a simple Monte Carlo approximation of $P(y_i|z,x)$ that uses a binary indicator $\mathbb{1}[y_i \in \mathcal{Y}_{z,x}]$ of whether or not antecedent y_i is in $\mathcal{Y}_{z,x}$ (the set of generated antecedents given prompt z and x):

$$\begin{aligned} P(y_i|x) &= \sum_z P(y_i|z,x)P(z|x) \\ &\approx \sum_z \mathbb{1}[y_i \in \mathcal{Y}_{z,x}]P(z|x) \end{aligned}$$

In §4, we show empirically that when using hundreds of prompts, MICE-SAMPLING is actually a better approximation than the LM probability of the first token (Zhao et al., 2021). In addition, MICE-SAMPLING is simpler to implement, as it does not require storing output logits and computing the softmax to obtain first token probabilities.

	Train (full)	**Dev-64**	**Dev-256**	**Test**
# anaphors	4,766	64	256	898
# antecedents	21,673	293	996	4,016
# documents	856	11	52	166
# sentences	5,833	76	346	1,066
# tokens	984,032	10,150	45,825	140,614

Table 1: Statistics of selected CHEMU-REF splits. Details on few-shot train sets are shown in Table 9 in the Appendix.

4 Experimental Setup

We evaluate our approach on CHEMU-REF, a corpus of synthetic procedures from chemical patents that are annotated with coreference and anaphora (Fang et al., 2021). The CHEMU-REF annotations contain fine-grained chemistry-specific relations such as TRANSFORMED, which indicates the mixture components undergo a chemical transformation, or WORK-UP, which represents a combination of compounds to isolate or purify a reaction product. In this work, we focus on modeling the general structure of anaphora in scientific protocols and understanding which compounds are combined at each step of the procedure, which does not require making these more fine-grained distinctions. Therefore, we pre-processed the data by collapsing all one-to-many CHEMU-REF relations into a single MULTIPLE-ANTECEDENT relation. We remove anaphors comprising compounds described with IUPAC nomenclature (Skonieczny, 2006), for example *"2,2,6,6-tetramethylpiperidine"*, while retaining nominal anaphors, such as *"the mixture"*, *"the solution"* and *"the reaction"*, which make up 90% of anaphors in the CHEMU-REF corpus. For train/dev/test splits, we use the original train split for training and development, and the original dev split for evaluation.[4] Following Gao et al. (2021), for each k-shot experiment, we sample five different training sets from the full training split using different seeds and report the mean. For model development and ablation studies, we use a small development set of 64 examples (Dev-64) to simulate a true few-shot learning setting. For evaluation on held-out data, we use the full CHEMU-REF development set as test data (Test) as well as a subsampled version with 256 examples (Dev-256) to control for high GPT-J inference costs. Selected dataset statistics are shown in Table 1.

We use F_1 as our evaluation metric. In particular, we compute the micro-F_1 between the predicted and gold sets of antecedents of the evaluation data. A predicted and a gold antecedent are considered the same if they are an exact match.

4.1 Implementation Details

Models We use GPT-J-6B (Wang and Komatsuzaki, 2021) as the backbone language model for MICE and in-context baselines since it was the largest publicly available autoregressive language model at the time we started this work.[5] Compared with other similar language models like GPT-2, GPT-J has a larger maximum sequence length of 2048. It is also pre-trained on the Pile (Gao et al., 2020), which covers in-domain data including chemical patents from USPTO[6] and PubMed articles. Similarly, we choose PROCBERT (Bai et al., 2021) for the student model in knowledge

[4]The official CHEMU-REF test set is hidden at `http://chemu2021.eng.unimelb.edu.au/`.

[5]Larger models, such as OPT-175B (Zhang et al., 2022), have become available recently.

[6]`https://www.uspto.gov/`

distillation due to its in-domain pre-training on synthetic procedures.

MICE We ensemble up to 256 prompts, with 2 or 5 in-context demonstrations in the prompt, for all few-shot settings. To calculate the similarity score $s(x, u)$ between the input x and a demonstration example u, we use SBERT model (Reimers and Gurevych, 2019)[7] with the `roberta-large` checkpoint since this model is widely used for measuring text similarity (Wang et al., 2020). We also experimented with calibrating the language model to be bias-free by applying the calibration procedure described in Zhao et al. (2021); however, we found that it did not improve performance in the context of anaphora resolution in scientific protocols.[8]

Antecedent Filtering To further improve the quality of predicted antecedents, we apply several post-processing rules that were developed on the Dev-64 split. Namely, we (1) filtered out all predictions that exceed a threshold length of 250 tokens (2) merged the antecedents that are sub-strings to the longest predicted antecedents (e.g. *"CH2CL2"* is merged into *"CH2CL2 (40 mL)")* and (3) filtered out all the antecedents with probability $P(y_i|z, x)$ in Eq. 2 less than 0.02 and probability $P(y_i|x)$ in Eq. 1 less than 0.1.

Knowledge Distillation To produce a compact model for inference, we perform knowledge distillation (Sanh et al., 2019; Jiao et al., 2020) via self-training, where the student model is trained on pseudo labels generated MICE. We experiment with three few-shot settings $k \in \{8, 32, 64\}$. In each setting, out of five training runs, we select the MICE-SAMPLING-$\{2\}$ model checkpoint (2 in-context examples) that achieves the median performance on the Dev-256 set as the teacher model. To train the student model (Schick and Schütze, 2021b; Lang et al., 2022), we randomly sample real unlabeled synthetic protocols from the chemical patent corpus collected in Bai et al. (2021). We then run rule-based anaphor detection (Appendix A) to identify anaphors within the sampled documents. Subsequently, the teacher model is used to predict antecedents. Finally, we train the PROCBERT student model in a two-stage process. We first fine-tune PROCBERT on M pseudo-labeled examples ($M \in \{50, 100, ..., 2000\}$) for 50 epochs, then further fine-tune it on the k examples with gold labels for 200 epochs. For the student model, we frame antecedent resolution as a sequence-labeling problem by transferring span-level antecedent labels into token-level labels with a BIO tagging scheme. We train the student model for token-level classification, where the input anaphor is marked with two special tokens `[Ana-start]` and `[Ana-end]`.

Computational Cost In addition to performance, we also measure the computational cost of the teacher and student models in knowledge distillation. Concretely, we measure floating point operations (FLOPs) of the two models for both training and inference using the FLOPs-counting code provided in Clark et al. (2020)[9]. Note that, the training of the student model requires M pseudo-labeled examples, so the training FLOPs of the student model will also include the FLOPs of generating those pseudo labels using the teacher model.

For all GPT-J-based experiments, generation is performed using greedy decoding up to a maximum of 256 tokens on 48GB A40 GPUs. When using 256 in-context experts, about three anaphors can be resolved per GPU hour.

4.2 Baselines

We compare MICE to the fine-tuning and in-context learning baselines described below. We use the k-shot training set to train the fine-tuning models or select in-context demonstrations for the in-context models. For all baselines, we use the development set Dev-64 for model selection and report the results on the held-out Dev-256 and test sets. Details on baseline implementation can be found in Appendix C.

E2E (Fang et al., 2021): We train this end-to-end neural anaphora resolution model developed for chemical procedures with minimal adaptations. [10]

PROCBERT (Bai et al., 2021): The student model in knowledge distillation (§4.1). For baseline comparison, we fine-tune only on gold data.

T5/T0-3B (Raffel et al., 2020; Sanh et al., 2022): We fine-tune T5-3B (3 billion parameters) and T0-

[7]`https://www.sbert.net/docs/pretrained_models.html`

[8]When prompted with content-free inputs, such as "N/A", the language model already generates bias-free answers (e.g. the LM generates "N/A" given "N/A" as a test input) without calibration, suggesting there is no significant bias towards specific answers for anaphora resolution.

[9]`https://github.com/google-research/electra/blob/master/flops_computation.py`

[10]`https://github.com/biaoyanf/ChEMU-Ref`

Model	4-shot	8-shot	16-shot	32-shot	64-shot	full
E2E (Fang et al., 2021)	0.9$_{1.1}$	10.3$_{5.3}$	31.6$_{8.1}$	42.5$_{5.4}$	51.9$_{2.3}$	77.4
ProcBERT (Bai et al., 2021)	20.3$_{7.9}$	27.1$_{3.7}$	36.1$_{4.1}$	45.3$_{6.7}$	55.0$_{4.4}$	**87.7**
T0-3B (Sanh et al., 2022)	23.5$_{7.0}$	30.8$_{4.3}$	42.1$_{2.3}$	49.5$_{1.3}$	58.1$_{3.2}$	83.2
T5-3B (Raffel et al., 2020)	27.3$_{3.7}$	33.6$_{3.9}$	42.5$_{6.7}$	55.6$_{6.2}$	**61.0$_{1.1}$**	83.8
Kate (Liu et al., 2022b)	36.2$_{9.6}$	44.7$_{5.0}$	46.8$_{2.9}$	46.4$_{3.9}$	48.4$_{1.5}$	-
Kate+ (Liu et al., 2022b)*	34.3$_{3.6}$	40.0$_{0.0}$	47.7$_{0.7}$	51.0$_{0.4}$	51.5$_{0.0}$	-
Product (Min et al., 2022a)	36.7$_{10.5}$	39.4$_{8.6}$	41.3$_{5.3}$	42.4$_{2.3}$	45.1$_{2.7}$	-
Mice-{2}	44.4$_{6.2}$	49.8$_{5.3}$	52.9$_{3.8}$	54.6$_{2.1}$	54.5$_{3.1}$	-
Mice-Sampling-{2}	**44.7$_{5.8}$**	50.6$_{5.4}$	55.1$_{2.9}$	56.8$_{3.0}$	59.0$_{3.1}$	-
Mice-{5}	44.5$_{6.5}$	**52.6$_{5.0}$**	55.8$_{4.4}$	57.4$_{1.2}$	58.4$_{1.9}$	-
Mice-Sampling-{5}	44.4$_{6.5}$	52.5$_{5.0}$	**56.8**$_{4.0}$	**57.5**$_{3.4}$	59.2$_{2.1}$	-

Table 2: Results on Dev-256. Bracketed numbers indicate the maximum number of demonstrations per prompt. For instance, Mice-{2} represents a mixture of $\min(k^2, 256)$ prompted language models where each prompt encodes two training examples. For each k-shot experiment, we report the mean and standard deviation across five different training set samples. Models marked with an asterisk * are reported with less than five trials, due to time/resource constraints. The **full** ChEMU-Ref training split contains 4,766 examples. Results on the test set are presented in Table 3.

3B for antecedent resolution using a QA prompt similar to Mice.

Kate (Liu et al., 2022b): For this model, each test example is associated with a single prompt constructed using K-nearest neighbors. For the sentence encoder, we use a pre-trained RoBERTa large model (`all-roberta-large-v1`) from the SBERT library (Reimers and Gurevych, 2019).

Kate+ Given a single prompt constructed by Kate, we sample 256 antecedent lists using nucleus sampling (Holtzman et al., 2020) and ensemble the predictions in a similar manner to Mice. This baseline ensembles the same number of predictions as Mice, with the difference being they are all sampled from the language model when conditioned on a single prompt.

Product (Min et al., 2022a): We adapt this ensemble-based demonstration method for text classification by first obtaining the output probabilities k times (one training example per prompt) and then computing the conditional probability of the antecedents given the input by taking the product of the aforementioned probabilities (i.e. $P(y|x) = \prod_z P(y|z, x)$).

5 Results and Analysis

Results on Dev-256 are presented in Table 2. We observe that variations of Mice either outperform (4-shot, 8-shot, 16-shot, 32-shot) or are competitive (64-shot) with the baselines. Furthermore, we found that while T5-3B works well for 64-shot, it still trails behind Mice and Mice-Sampling (and even Kate) when using fewer labeled data.[11] Ad-

Model	8-shot	32-shot	64-shot	full
E2E (Fang et al., 2021)	9.9	42.3	51.8	75.8
ProcBERT (Bai et al., 2021)	30.8	41.9	57.6	**78.3**
T5-3B (Raffel et al., 2020)	33.9	52.0	58.6	72.6
Kate (Liu et al., 2022b)	37.9	38.6	42.3	-
Product (Min et al., 2022a)	34.3	38.5	49.0	-
Product-{2}	45.6	51.1	51.3	-
Mice-{2}	48.1	52.6	53.9	-
Mice-Sampling-{2}	48.5	53.9	55.7	-
Know. Distill. (2000 exam.)	**56.2**	**59.4**	**64.6**	-

Table 3: F_1 on the full test set. For the knowledge distillation model, we picked the median trial from Mice-Sampling-{2} of Table 2 to run experiment on, due to the expensive inference cost on large amount of unlabeled data. For others, we averaged over five training samples.

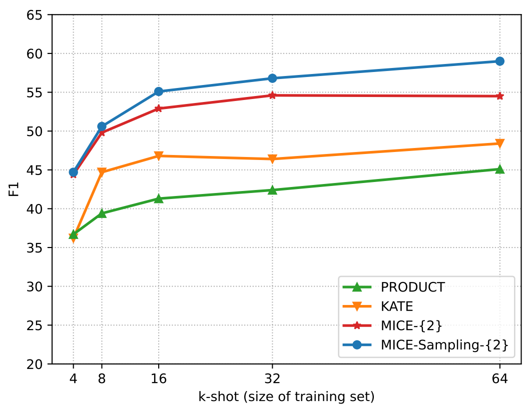


Figure 4: Mean F_1 of in-context learning models vs. training set size k, on Dev-256.

[11]We also experimented with T-few (Liu et al., 2022a), a performant T0-based parameter-efficient fine-tuning approach.

ditionally, we observe that MICE outperforms other in-context learning models that use a single most performant prompt (KATE) or a single training example per prompt (PRODUCT). This is highlighted in Figure 4, where we plot the performance of various in-context learning models as a function of training set size k.

Results on the full test set (Table 3) shows similar trends. We also experiment with extending the PRODUCT baseline to include two in-context demonstrations per prompt (PRODUCT-{2}). While increasing the number of in-context demonstrations in PRODUCT also boosts the performance for all k-shot, MICE and MICE-SAMPLING still outperform PRODUCT-{2}.

KATE+ vs. MICE We highlight the results of KATE+ in comparison to KATE and MICE. KATE+ performs somewhat better than KATE, likely due to state-of-the-art text generation methods (e.g. nucleus sampling) as well as ensembling independently-sampled sequences conditioned on a single prompt. However, MICE significantly outperforms KATE+, indicating the importance of ensembling prompts with different demonstrations and permutations.

Knowledge Distillation The impact of knowledge distillation is explored in Table 4. We observe that with only 50 pseudo-labeled examples, the student model outperforms the PROCBERT baseline by 8.0 F_1 for 8-shot, 5.1 F_1 for 32-shot and 1.7 F_1 for 64-shot, showing the effectiveness of distillation. The performance of the student model improves as the number of pseudo-labeled examples increases. With 2000 pseudo-labeled examples, the inference-efficient student model (110M parameters) outperforms the teacher model MICE-SAMPLING. We hypothesize that the student model outperforms the teacher because after training on pseudo-labels generated by MICE, the student model is further fine-tuned on the k available gold-labeled examples, as described in §4.1. This approach combines the benefits of in-context learning with fine-tuning achieving better performance than either in isolation. In-context learning does not incur any training costs, but the number of FLOPs required by MICE-SAMPLING for inference is roughly 1,500 times the computational cost of the student model.

With suggested hyper-parameters, T-few performs worse than the full-model fine-tuning of T0-3B on anaphora resolution, so we leave the further exploration of T-few to future work.

Model	F_1 8-shot	F_1 32-shot	F_1 64-shot	Train FLOPs	Infer FLOPs (1000 exam.)
PROCBERT	28.0	41.1	56.7	3.2e15	1.2e14
Teacher	54.1	55.7	59.9	0	1.9e17
Student (# pseudo labels)					
- 50	36.0	46.2	58.4	1.4e16	1.2e14
- 100	43.5	49.9	62.2	2.4e16	1.2e14
- 200	42.1	52.3	60.4	4.5e16	1.2e14
- 500	49.1	52.2	63.3	1.1e17	1.2e14
- 1000	50.1	51.8	64.1	2.1e17	1.2e14
- 2000	**54.3**	**55.8**	**64.3**	4.1e17	1.2e14

Table 4: Dev-256 F_1 and training/inference FLOPs of the teacher model (MICE-SAMPLING) and the student model in knowledge distillation. The training FLOPs of the student model under different few-shot settings are almost the same. With 2000 pseudo-labeled examples, the student model (110M parameters) can match or outperform the teacher model in terms of F_1. Although inference FLOPs of the teacher model are around 1500 times the FLOPs of the student model, considering the cost of training the student model to match the teacher model's performance, the teacher model is actually more cost effective than the student model for inference when the number of inference examples is low, e.g., less than 2100 for 32-shot. For rows where the number of FLOPs varies depending on the number of training examples, we show the maximum.

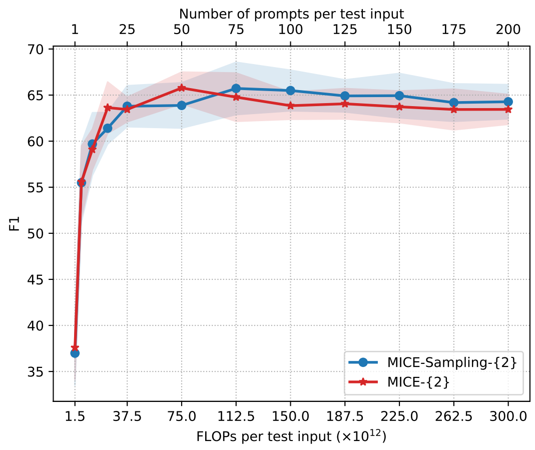


Figure 5: Mean F_1 ($\pm$ one std. dev.) vs. the number of prompts used in MICE. Performance plateaus after the number of experts is increased to 75, suggesting an appropriate choice for the minimum number of prompts in this context. The number of prompts corresponds to the number of inference passes, which we measure using FLOPs per test example (§4.1).

Number of mixture prompts vs. Performance We explore the effect of varying the number of mixture prompts in Figure 5. While adding more prompts leads to rapid F_1 improvement, the performance plateaus after about 75 prompts. This suggests an appropriate minimum number of experts to use in MICE.

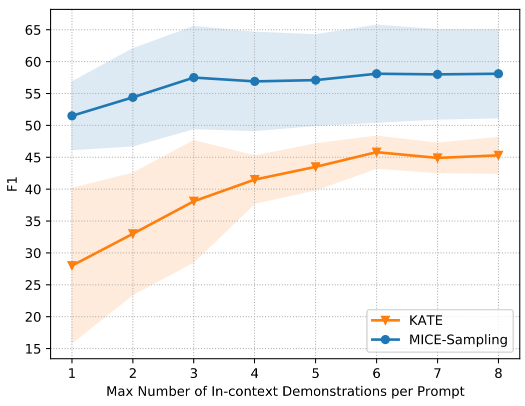


Figure 6: Mean F_1 ($\pm$ one std. dev.) vs. max number of in-context demonstrations per prompt. We observe a large performance gap between MICE-SAMPLING and KATE.

Number of in-context demonstrations vs. Performance Figure 6 shows the effect of varying the number of in-context demonstrations per prompt, for mixture models (MICE-SAMPLING) and single-prompted models (KATE). As the number of in-context demonstration per prompt increases, KATE F_1 steadily increases. This effect is less pronounced in MICE, where adding more than three in-context demonstrations per prompt does not significantly improve performance. However, there is still a large performance gap between MICE and KATE, regardless of the number of in-context demonstrations.

6 Related Work

Ensembling prompts Recent work has explored different strategies for ensembling LM predictions. Lester et al. (2021) and Zhao et al. (2022) aggregate predictions from multiple prompts via either majority voting or weighting on development set performances. Jiang et al. (2020) and Qin and Eisner (2021) ensemble a small number of prompt templates or soft prompts, then learn their weights from a large training set. Asai et al. (2022) combines soft prompts pre-trained on multiple source tasks with fine-tuned attention weights to generate prompts for a target task. In contrast to all the above work, MICE generates a large number of prompts from a small number of training examples and combines their predictions using mixtures of experts with similarity-based gating.

Similar to our work, Min et al. (2022a) shows that ensembling the predictions of k one-shot prompts by multiplying their LM probabilities is better than concatenating all k examples as one prompt, which we use as a baseline (PRODUCT). Concurrently, Lang et al. (2022) demonstrates the advantage of ensembling k one-shot prompts via co-training the ensembling parameters using a smaller model. However, they train the ensembling parameters with an MLP layer and thus require all training examples to share the same label space, which is not applicable to anaphora resolution. Importantly, both Min et al. (2022a) and Lang et al. (2022) only consider a small number of prompts, whereas we demonstrate the benefits of generating and combining hundreds of prompts as a mixture-of-experts.

Few-shot anaphora resolution There is very little prior work on few-shot coreference or anaphora resolution. Prior work (Perez et al., 2021; Min et al., 2022b) has applied in-context learning to the benchmark datasets Winograd Schema Challenge (WSC) (Levesque et al., 2012) and WinoGrande (Sakaguchi et al., 2019). Both datasets are intended to be an alternative to the Turing Test. As such, they contain short, syntactically simple sentences, and require a different type of world knowledge to resolve references than those seen in scientific protocols. Agrawal et al. (2022) demonstrates the effectiveness of zero-shot prompting for coreference resolution on clinical data. They introduce task-specific programs (called *resolvers*) that map language model outputs to discrete label space, which can be used in conjunction with MICE for further improvements on few-shot coreference resolution.

Mixture-of-Experts in Language Modelling Mixture-of-Experts (MoE) based language models have been shown to improve performance and efficiency across a variety of NLP tasks (Shazeer et al., 2017; Du et al., 2022; Gururangan et al.). These models were pre-trained on a mixture of datasets with different domains, either via learning the gating weights at a token level (Shazeer et al., 2017; Du et al., 2022) or document level (Gururangan et al.). Unlike prior work, MICE combines the predictions of many in-context experts, each of which encodes a permutation of demonstrations drawn from a handful of training examples. The gating weights in MICE are computed based on similarity scores between test input and in-context examples.

7 Conclusion

In this paper, we propose and demonstrate the effectiveness of Mixtures of In-Context Experts for few-shot anaphora resolution in chemical synthesis protocols. MICE generates a large number of in-context experts (prompts) from a few training examples, where each expert consists of randomly permuted demonstrations. Predictions made by these experts are combined in a mixture model with a similarity-based gating function. Our experiments show that MICE significantly improves the performance of in-context learning for anaphora resolution in scientific protocols, using just a handful of training examples. We further demonstrate that knowledge distillation can dramatically reduce the costs of inference while maintaining performance, which increases MICE's potential applicability in working systems.

Limitations

A challenging bottleneck for MICE is its expensive inference cost: despite the promising few-shot capabilities, the inference cost is even more expensive when using a language model with over 6 billion parameters. Although we show that knowledge distillation can address this problem, future work may further investigate how to improve smaller models' in-context capabilities.

Acknowledgements

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001119C0108, in addition to the NSF (IIS-2052498) and IARPA via the BETTER program (2019-19051600004). The views, opinions, and/or findings expressed are those of the author(s) and should not be interpreted as representing the official views or policies of the Department of Defense, IARPA or the U.S. Government. This work is approved for Public Release, Distribution Unlimited.

References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are zero-shot clinical information extractors. *arXiv preprint arXiv:2205.12689*.

Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. Attentional mixtures of soft prompt tuning for parameter-efficient multi-task knowledge sharing. *arXiv preprint arXiv:2205.11961*.

Fan Bai, Alan Ritter, and Wei Xu. 2021. Pre-train or annotate? Domain adaptation with a constrained budget. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*.

Jaemin Cho, Minjoon Seo, and Hannaneh Hajishirzi. 2019. Mixture content selection for diverse sequence generation. 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)*.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Doug Downey, Oren Etzioni, and Stephen Soderland. 2005. A probabilistic model of redundancy in information extraction. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.

Biaoyan Fang, Christian Druckenbrodt, Saber A Akhondi, Jiayuan He, Timothy Baldwin, and Karin Verspoor. 2021. ChEMU-ref: A corpus for modeling anaphora resolution in the chemical domain. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*.

Suchin Gururangan, Mike Lewis, Ari Holtzman, Noah A. Smith, and Luke Zettlemoyer. DEMix layers: Disentangling domains for modular language modeling. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International Conference on Machine Learning*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural computation*.

Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. 2018. An annotated corpus for machine reading of instructions in wet lab protocols. In *Proceedings of NAACL-HLT*.

Hunter Lang, Monica Agrawal, Yoon Kim, and David A. Sontag. 2022. Co-training improves prompt-based learning for large language models. *CoRR*.

Alexander J Lawson, Jürgen Swienty-Busch, Thibault Géoui, and David Evans. 2014. The making of reaxys—towards unobstructed access to relevant chemistry information. In *The Future of the History of Chemical Information*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The Winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022a. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022b. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Sewon Min, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022a. Noisy channel language model prompting for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022b. MetaICL: Learning to learn in context. In *NAACL-HLT*.

Sheshera Mysore, Zachary Jensen, Edward Kim, Kevin Huang, Haw-Shiuan Chang, Emma Strubell, Jeffrey Flanigan, Andrew McCallum, and Elsa Olivetti. 2019. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop*.

Silviu Paun, Juntao Yu, Nafise Sadat Moosavi, and Massimo Poesio. 2022. Scoring coreference chains with split-antecedent anaphors. *arXiv preprint arXiv:2205.12323*.

Ethan Perez, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. In *Advances in Neural Information Processing Systems*.

Guanghui Qin and Jason Eisner. 2021. Learning how to ask: Querying LMs with mixtures of soft prompts. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. WinoGrande: An adversarial Winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*.

Katharine Sanderson. 2019. Automation: Chemistry shoots for the moon. *Nature*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc V. Le, Geoffrey E. Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *5th International Conference on Learning Representations, ICLR*.

Stanislaw Skonieczny. 2006. The IUPAC rules for naming organic molecules. *Journal of chemical education*.

Ronen Tamari, Fan Bai, Alan Ritter, and Gabriel Stanovsky. 2021. Process-level representation of scientific protocols with interactive annotation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*.

Hardik Vala, Andrew Piper, and Derek Ruths. 2016. The more antecedents, the merrier: Resolving multi-antecedent anaphors. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

Alain C Vaucher, Philippe Schwaller, Joppe Geluykens, Vishnu H Nair, Anna Iuliano, and Teodoro Laino. 2021. Inferring experimental procedures from text-based representations of chemical reactions. *Nature communications*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, et al. 2020. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020*.

Bonnie Webber and Breck Baldwin. 1992. Accommodating context change. In *30th Annual Meeting of the Association for Computational Linguistics*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*.

Wei Wu, Fei Wang, Arianna Yuan, Fei Wu, and Jiwei Li. 2020. CorefQA: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Xiaohan Yang, Eduardo Peynetti, Vasco Meerman, and Chris Tanner. 2022. What GPT knows about who is who. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP*.

Juntao Yu, Nafise Sadat Moosavi, Silviu Paun, and Massimo Poesio. 2020. Free the plural: Unrestricted split-antecedent anaphora resolution. In *Proceedings of the 28th International Conference on Computational Linguistics*.

Michelle Yuan, Patrick Xia, Chandler May, Benjamin Van Durme, and Jordan Boyd-Graber. 2022. Adapting coreference resolution models through active learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel

Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: open pre-trained transformer language models.

Mengjie Zhao, Fei Mi, Yasheng Wang, Minglei Li, Xin Jiang, Qun Liu, and Hinrich Schütze. 2022. LM-Turk: Few-shot learners as crowdsourcing workers in a language-model-as-a-service framework. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *ICML*.

A Anaphor Detection

We develop a rule-based system using simple string matching patterns. We developed the system using the full train set, optimized on the Dev-64 development set, and evaluated on Dev-256 and Test set. Anaphora in synthetic protocols are expressed using relatively consistent and easy to identify expressions, for example: *"the mixture", "the reaction solution"*, and *"the resulting solution"*, making them relatively easy to identify using a simple approach using regular expressions. As a result, our system can effectively identify most of these spans, achieving high F_1 scores when evaluated on different evaluation splits (Table 5). An example of a pattern that matches all anaphors in Figure 1 test input is shown here: `https://regex101.com/r/pImmBT/1`

	P	**R**	$\mathbf{F_1}$
Dev-64	98.5	100	99.2
Dev-256	97.6	95.7	96.6
Test	95.6	85.5	90.3

Table 5: Results of our rule-based anaphor detection system on different evaluation splits.

B MICE Implementation Details

We present the hyperparameters for MICE (as well as other in-context learning baselines) in Table 7. We also experimented with different prompt ordering of in-context demonstrations Liu et al. (2022b) and contextual calibration Zhao et al. (2021). All ablation experiments were evaluated on Dev-64 development set.

Ordering of In-context Demonstrations Since different prompt ordering can play a significant role in the performance, we measure the effect of different in-context ordering on KATE. Given a test input, we first select the most similar examples using based on the cosine similarity between the test input and the example embeddings (encoded using pre-trained RoBERTa large model). We then consider the following ordering:

- ASCEND: Sorting the order of in-context demonstrations from least to most similar examples (i.e. most similar example closest to test input)
- DESCEND: Reverse of ASCEND
- MIXED: Random shuffle of in-context demonstrations

The results on Table 6 show there is not much difference between different orderings. This result is in line with the findings in **?** that prompt ordering is dataset-dependent. As such, we select ASCEND as the de facto ordering for all our in-context learning experiments.

Contextual Calibration Zhao et al. (2021) shows that prompted LMs can be biased towards specific outputs, regardless of the test input. We thus experimented with the calibration procedure for generation tasks described in Section 5 of Zhao et al. (2021), also referencing their code.[12] Specifically, we computed calibration parameters using the output probabilities of the content-free prompt (e.g. prompt with "N/A" test input), and used these parameters to update output probabilities when prompted with actual test inputs. However, we found that applying this calibration procedure does not improve performance. We further observe that when prompted with content-free inputs, the language model already generates bias-free answers (e.g. generates "N/A" given "N/A" test input) without calibration. This suggests there is no significant bias towards specific answers for the case of split-antecedent resolution, whereas current calibration techniques are designed for cases where the LM generates biased answers given content-free inputs, (e.g. generates a label given "N/A" test input).

C Baseline Implementation Details

For E2E (Fang et al., 2021), we run the TensorFlow code provided in the original paper (following the same hyperparameters) to get the model performance under our few-shot settings. Other baselines, including PROCBERT, T5-3B, and T0-3B, are implemented using Huggingface Transformers (Wolf et al., 2020). The hyperparameters used for these baselines are shown in Table 8.

[12] `https://github.com/tonyzhaozh/few-shot-learning`

	4-shot	8-shot	16-shot	32-shot	64-shot
ASCEND	$34.3_{4.9}$	$45.3_{2.8}$	$\mathbf{46.8}_{5.1}$	$\mathbf{46.8}_{6.6}$	$\mathbf{46.5}_{5.9}$
DESCEND	$\mathbf{34.8}_{4.8}$	$\mathbf{46.2}_{1.3}$	$45.1_{4.8}$	$44.6_{6.6}$	$46.5_{4.7}$
MIXED	$34.1_{7.6}$	$43.9_{3.3}$	$46.1_{4.7}$	$46.2_{4.3}$	$46.3_{4.1}$

Table 6: Order of in-context demonstration results.

Hyperparameter	KATE	KATE+	PRODUCT	MICE-{2} MICE-SAMPLING-{2}	MICE-{5} MICE-SAMPLING-{5}
# prompts per example	1	1	k	$\min(256, k^2)$	$\min(256, k^2)$
Max # in-context per prompt	$\min(k, d_x)$	$\min(k, d_x)$	1	2	5
Order of in-context	ASCEND	ASCEND	-	ASCEND	ASCEND
$P(y_i\|z, x)$ threshold	0.0	0.02	0.02	0.02	0.02
$P(y_i\|x)$ threshold	0.0	0.1	0.1	0.1	0.1
top-k	-	50	-	-	-
top-p	-	0.95	-	-	-

Table 7: Hyperparameters for in-context learning models. k is the size of training set (k-shot). d_x depends on the length of test input x and retrieved in-context examples (Figure 3). ASCEND denotes sorting the order of in-context demonstrations from least to most similar examples (i.e. most similar example closest to test input). Top-k and top-p are hyperparameters for nucleus sampling used for KATE+.

Hyperparameter	PROCBERT	T5-3B / T0-3B
# epochs	200	20
batch size	32	4
learning rate	2e-5	1e-4
max. encoder length	512	512
max. decoder length	-	256
optimizer	AdamW	AdamW

Table 8: Hyperparameters for PROCBERT, T5-3B, and T0-3B.

Data Split	4-shot	8-shot	16-shot	32-shot	64-shot	Train (full)	Dev-64	Dev-256	Test
# anaphors	4	8	16	32	64	4,766	64	256	898
# antecedents	13.4	24.2	61.2	132.8	266.0	21,673	293	996	4,016
# documents	2.2	3.8	6.0	9.6	17.6	856	11	52	166
# sentences	8.0	15.2	27.0	50.4	92.2	5,833	76	346	1,066
# tokens	445.8	939.6	2,257.6	4,665.2	9,775.8	984,032	10,150	45,825	140,614

Table 9: Dataset splits details. The k-shot dataset statistics are averaged over five random samples.