# Prefix Tuning: Optimization of Large Natural Language Models

**Grace Wang**
Princeton University
gw17@princeton.edu

**Nobline Yoo**
Princeton University
nobliney@princeton.edu

**Richard Zhu**
Princeton University
ryzhu@princeton.edu

## Abstract

Pretrained language models are models whose weights have been optimized to learn general language structure and syntax. Traditionally, fine-tuning has been used to repurpose large pretrained models for specific tasks and involves adjusting all model-level parameters. This means that researchers wanting to run a model with $n$ parameters on $m$ separate tasks have to store $O(mn)$ parameters, which is a significant memory cost across tasks. In addition, even on an individual task level, the large number of optimizable parameters incurs significant training costs—take for example GPT-2, which has 345 million parameters—in both compute time and memory.

In our work, we seek to reproduce the ACL conference paper "Prefix-Tuning: Optimizing Continuous Prompts for Generation." In this paper, the authors propose an alternative to fine-tuning: prefix tuning. They propose a baseline and experiment with several ablations to stress-test the model in various environments—e.g. contrasting low-data (few training examples) and high-data (many training examples) performance, and exploring performance differences between continuous and discrete embeddings.

(Li and Liang, 2021) propose fixing all language model parameters and optimizing only a small set of "continuous task-specific vectors" that are appended to the model. In contrast to fine-tuning, this means that researchers running a model with $n$ frozen parameters on $m$ tasks with $p << n$ optimizable parameters in the prefix have only to store $n + m * p$ parameters. (Li and Liang, 2021) show that with "only 0.1% of the parameters, prefix-tuning obtains comparable performance in the full data setting." This extrapolates to storing $n(1 + 0.001m)$, as compared with $nm$, parameters.

By solving for when $1 + 0.001m = m$, we can estimate a threshold for the number of tasks $m$ for which the memory cost (number of parameters to store) is the same for fine-tuning versus prefix tuning. This threshold $m$ is 1.001, which means that for more than 1 task, prefix tuning is more space-efficient.

Hence, prefix-tuning is worth exploring further, and in our project, we (1) successfully reproduce the table-to-text generation baseline of GPT-2 on the E2E dataset, and (2) propose four additional ablation studies: (i) modifying the prefix length, (ii) tuning GPT-2 with prefix and infix in conjunction, (iii) analyzing a new prefix initialization, and (iv) modifying the decode mode at evaluation time to analyze the impact of making changes at the testing stage on model performance.

## 1 Introduction

Deep natural language generation models have become increasingly effective, but come with large parameter spaces on the order of 175 billion (Brown et al., 2020). Retraining these models from the bottom-up for specific natural language tasks, such as table-to-text generation or text summarization, can be computationally expensive. Instead, researchers have relied on pre-trained models that are then tweaked to perform well on specific tasks; this is known as *fine-tuning* (Devlin et al., 2018; Radford et al.).

Given that these models have encoded high-level language patterns, the process of transfer learning is often quite effective given that the tasks reside in similar domains.

However, as models becomes larger and the number of parameters increases, fine-tuning can become computationally inhibitory. Prefix-tuning allows us to cater the model to specific tasks by appending a small vector to inputs and optimizing only on these small, fixed-length, continuous vectors to specific tasks (Figure 1).

This allows researchers to train task-specific models while only optimizing 0.1% of the total model parameters, as the authors did in (Li and

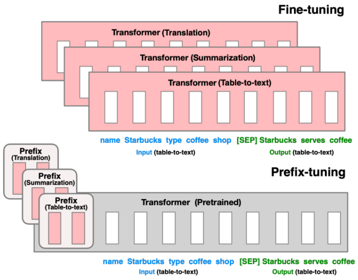
Fine-tuning
Transformer (Translation)
Transformer (Summarization)
Transformer (Table-to-text)
name Starbucks type coffee shop [SEP] Starbucks serves coffee
Input (table-to-text)
Output (table-to-text)
Prefix (Translation)
Prefix (Summarization)
Prefix (Table-to-text)
Prefix-tuning
Transformer (Pretrained)
name Starbucks type coffee shop [SEP] Starbucks serves coffee
Input (table-to-text)
Output (table-to-text)

Figure 1: Architectural difference between fine-tuning and prefix-tuning. Figure from (Li and Liang, 2021)

Liang, 2021). Prefix-tuning GPT-2$_{MEDIUM}$ on the table-to-text task on 0.1% of the total parameter space, despite the smaller number of affected parameters, outperforms fine-tuning on the DART metric by a measurable amount. Prefix tuning achieves 1.5, 9.3, and 0.2 point improvements on the BLEU score over fine-tuning on the E2E (Novikova et al., 2017a), WebNLG (Gardent et al., 2017), and DART datasets (Nan et al., 2021), respectively (Li and Liang, 2021).

## 2 Related Work

It is important to provide some scholarly context for our work in reproducing the baseline and proposing new ablations for prefix tuning, so we discuss one paper (Chen et al., 2022) that also revisited prefix tuning. In (Chen et al., 2022), the authors conduct an analytical review of various parameter-efficient tuning methodologies.

Specifically focusing on prefix tuning, they conduct 20 runs of training and evaluation with 20 different random seeds. Using this methodology, they find that prefix tuning is generally unstable, because different orderings of training data cause significant variance in the accuracy. Unlike (Li and Liang, 2021), (Chen et al., 2022) concludes that (1) prefix tuning consistently underperforms fine-tuning in low, medium, and high-data settings and that (2) previous papers came to different conclusions because they were picking the best-accuracy results over multiple runs, rather than evaluating holistically over all runs.

Our paper is similar to (Chen et al., 2022) in the spirit of revisiting prefix tuning, attempting to reproduce the baseline, and suggesting new angles from which to approach the original formulation.

E2E is a closed-domain (restaurant reviews) table-to-text generation dataset that consists of nearly 50,000 examples (Novikova et al., 2017a). Figure 2 illustrates a data instance from E2E. (Shen et al., 2019) provides the previous state of the art for the E2E dataset across BLEU, NIST, METEOR (MET), Rouge-L (R-L), and CIDEr metrics. However, (Li and Liang, 2021) exceeds performance across all five metrics on the E2E dataset, and for this paper we use their prefix-tuning metrics as the new state of the art (SOTA).

## 3 Approach

The central task explored in this work is table-to-text generation. This task involves taking input in the form of a table and generating textual content based on the table. We use the E2E dataset as our testbed.

To evaluate our models we mainly compare 3 scores: METEOR, ROUGE-L, and CIDEr. METEOR represents the harmonic mean of precision and recall. ROUGE-L measures the longest common subsequence between the model output and gold output. CIDEr computes the cosine similarity between target and output descriptions.

The main language model used throughout all the experiments was GPT-2, which is essentially a scaled up version of GPT. GPT-2 is a transformer based language model from OpenAI with 1.5 billion parameters and 48 layers, trained on a corpus of text from 8 million websites. As it is an attention model, it focuses on the previous most relevant words in order to predict the next token in a sequence. GPT-2 also uses task conditioning, which allows it to learn multiple tasks with the same underlying model. Through task conditioning, GPT-2 is able to learn new tasks based on a given instruction and with no provided examples, which is called zero-shot learning.

In reproducing the baseline, our goal is to validate the methodology and results of the original authors. And in proposing four new ablations, we aim to stress-test the original authors' formulation. Our first ablation is modifying the prefix length with prefix tuning on the E2E dataset. In the original paper, the authors perform experiments varying prefix length (0, 5, 10, 20, 40) on DART, a much larger, open-domain, table-to-text generation dataset. The central concern in this experiment is finding the optimal prefix length (i.e. the threshold prefix length after which accuracy declines). Typically, a larger prefix length leads to more expressivity in the model, because there are more optimizable parameters. However, there may be an optimal threshold to the prefix length, because after a certain point, the model overfits to the training data and loses generalizability to unseen, testing data.

In proposing our first new ablation, we seek to explore how prefix length affects accuracy and overfitting in a *closed-domain* dataset (E2E) that is about three-fifths the size of DART. For example, if we find that there is not a threshold prefix length after which accuracy declines, then this may suggest that in closed-domain datasets, overfitting due to large prefix lengths is not as big of a concern, because the training and testing set are inherently more similar in distribution (because they are closed-domain) than in open-domain datasets. On the other hand, if we find that there is a clear threshold prefix length, then this may suggest that in smaller datasets, overfitting due to large prefix lengths is still a concern, because there is a higher chance that the distributions between the training and testing set are different.

In proposing our second ablation, we seek to explore how employing both prefix and infix tuning can help or harm accuracy. We hypothesize that adding infix tuning in addition to prefix tuning will allow the model to be more powerful due to the increase in adjustable weights.

In proposing our third ablation, we seek to investigate the effect of initializing our prefix to a known value. (Li and Liang, 2021) performs basic experimentation across a few prefix initializations, among which "active," "elephant," and "table-to-text:" achieve the highest BLEU scores. They conclude that initializing with task-relevant words achieves higher performance. We investigate the effect of adding numbers to the prefix initialization, which has not been attempted previously, in order to search for an initialization that allows us to reach a lower local optimum more quickly. This ablation allows us to outperform (Li and Liang, 2021) on the table-to-text task (using the E2E training set) across METEOR and CIDEr metrics, so we set forth a new SOTA.

In proposing our fourth ablation, we explore how modifications to the original formulation at evaluation time might improve model performance, specifically testing sampling as an alternative decoding approach to beam search.

| **Flat MR** | **NL reference** |
|---|---|
| name[Loch Fyne],<br>eatType[restaurant],<br>food[French],<br>priceRange[less than £20],<br>familyFriendly[yes] | Loch Fyne is a family-friendly restaurant providing wine and cheese at a low cost. |
| | Loch Fyne is a French family friendly restaurant catering to a budget of below £20. |
| | Loch Fyne is a French restaurant with a family setting and perfect on the wallet. |

Figure 2: Example data from E2E dataset. Figure from (Novikova et al., 2017b)

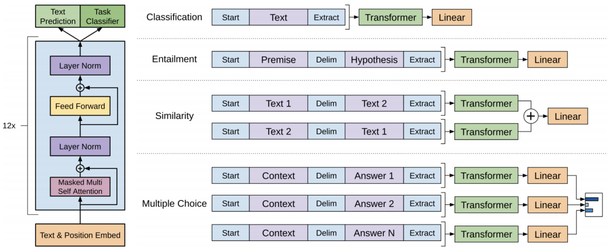

Figure 3: GPT architecture. Figure from (Radford et al., 2018)

### 3.1 Baseline

We emulate the baseline results using training only the prefix and also training only the infix. We achieve the results shown in Table 1 and 2. We achieve results on par with those of (Li and Liang, 2021). Our meteor score is 20 basis points (1 basis point or bp is equivalent to 0.01%) higher, our Rouge-L score is 90 bps lower, and our CIDEr score is 600 bps higher.

The baseline using prefix only was trained with the same model and hyperparameters as those used in (Li and Liang, 2021). We used a GPT-$2_{\text{MEDIUM}}$ model for table-to-text generation on the E2E dataset, 10 epochs, batch size 5, learning rate $5 \cdot 10^{-5}$, and prefix length 10. Training and evaluation took a combined 4.45 hours on a Tesla T4 GPU unit.

The baseline using infix only was trained 5

E2E (prefix-only)

| | **MET** | **R-L** | **CIDEr** |
|---|---|---|---|
| Li and Liang 2021 | 46.3 | **72.1** | 2.46 |
| Ours | **46.5** | 71.2 | **2.52** |

Table 1: Baseline results with prefix only

E2E (infix-only)

| | MET | R-L | CIDEr |
|---|---|---|---|
| Li and Liang 2021 | 45.8 | 69.9 | 2.40 |
| Ours | **46.0** | **70.5** | **2.41** |

Table 2: Baseline results with infix only

E2E (prefix vs infix baseline)

| | MET | R-L | CIDEr |
|---|---|---|---|
| Prefix-only | **46.0** | **71.0** | **2.455** |
| Infix-only | 46.0 | 70.5 | 2.41 |

Table 3: Our baseline prefix only vs infix only model - both trained with prefix length of 5

epochs, learning rate 0.00008, and prefix length 5. The first row in Table 2 shows the original author's results training with infix only with prefix length of 10. Our results are comparable and even exceed their infix model despite using a smaller prefix length. Table 3 compares the results of our prefix-only versus infix-only model, both trained with a prefix length of 5. Similar to the original authors, we conclude that prefix tuning outperformms infix tuning, because the prefix is able to affect the activations of both the input and output, while the infix only affects that of the output.

We use beam search for decoding and a single prefix prepended to the input/infix appended to the $x$ in the input, with random initialization of the prefix/infix, respectively.

Additional detail on the process for achieving baseline metrics on par with those provided in the paper are shown below.

For the first two weeks, we devoted our time to getting the baseline running. We created a group Google account, where we purchased Google Colab Pro, so that all three group members could access and track progress on the Colab notebooks.

Initially, we cloned the original authors' github repository into our shared account Drive. We created missing files, and ran our model with the parameters suggested. Unfortunately, at evaluation time, the results returned were the same regardless of the training hyperparameters we used. So, we contacted the original author, after which we received a link to her CodaLab repository. Based on our exploration of the CodaLab files, we adjusted our local repository to include the CodaLab python scripts and DART. We were finally able to replicate close to the original baseline results.

**Prefix Length Ablation on E2E**

| Prefix Length | MET | R-L | CIDEr |
|---|---|---|---|
| 5 | 46.0 | 71.0 | 2.455 |
| 10 | 46.0 | 71.4 | 2.463 |
| 15 | 46.0 | 71.6 | 2.456 |
| 20 | 46.1 | 71.4 | 2.473 |
| 25 | 45.9 | 70.8 | 2.459 |
| 30 | 46.1 | 71.2 | 2.460 |

Table 4: Results from the prefix length modification ablation study

### 3.2 Ablation 1: Modify Prefix Length with Full Prefix-Tuning on E2E

In the baseline study, the authors use a prefix length of 10. The authors also perform experiments varying prefix length (0, 5, 10, 20, 40) on the DART dataset. On the E2E dataset, the authors perform experiments varying prefix length with only discrete embeddings of real words, rather than with "virtual tokens" which are continuous embeddings (as proposed in prefix tuning methodology).

In *our* ablation study, we experiment with varying prefix lengths using prefix tuning (continuous embeddings) and the E2E dataset, which is unique from the two ablations that the original authors proposed. Specifically, we seek to explore how prefix length affects accuracy and overfitting in a *closed-domain* dataset (E2E) that is smaller than DART.

We use prefix lengths of 5, 10, 15, 20, 25, and 30. A shorter prefix length may be faster to train but corresponds to fewer trainable parameters and is thus limited in the amount of expressiveness. A longer prefix length may take longer to train but has more parameters to train, which may result in more expressivity. Our results paper find that with E2E, substantial expressiveness can be captured even with just a short prefix of length 5. We find that accuracy peaks around prefix length of 15 or 20, depending on the metric being evaluated (ROUGE-L in the former, METEOR and CIDEr in the latter). Cutting the original prefix length in half to five slightly reduces performance for Rouge_L and CIDEr but maintains the same METEOR score.

### 3.3 Ablation 2: Using Prefix and Infix in Conjunction

In the original paper, the authors test two placements of the trainable activation: (1) [trainable activate (prefix); $x$; $y$] (Figure 4) and (2) [$x$; trainable activation (infix); $y$] (Figure 5). They find that infix-tuning underperforms prefix-tuning, because the infix only affects the activation of $y$, whereas the prefix affects the activations of both $x$ and $y$.

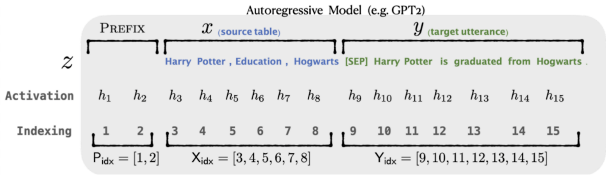
Autoregressive Model (e.g. GPT2)
PREFIX
$x$ (source table)
$y$ (target utterance)
$z$
Harry Potter , Education , Hogwarts [SEP] Harry Potter is graduated from Hogwarts .
Activation
$h_1$ $h_2$ $h_3$ $h_4$ $h_5$ $h_6$ $h_7$ $h_8$ $h_9$ $h_{10}$ $h_{11}$ $h_{12}$ $h_{13}$ $h_{14}$ $h_{15}$
Indexing
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
$P_{idx} = [1,2]$
$X_{idx} = [3,4,5,6,7,8]$
$Y_{idx} = [9,10,11,12,13,14,15]$

Figure 4: Prefix-only architecture. Figure from (Li and Liang, 2021)

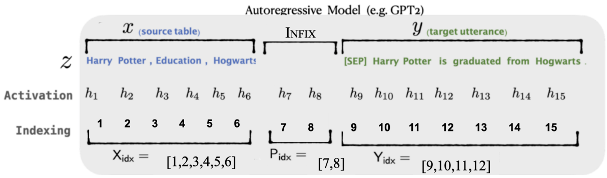
Autoregressive Model (e.g. GPT2)
$x$ (source table)
INFIX
$y$ (target utterance)
$z$
Harry Potter , Education , Hogwarts
[SEP] Harry Potter is graduated from Hogwarts .
Activation
$h_1$ $h_2$ $h_3$ $h_4$ $h_5$ $h_6$ $h_7$ $h_8$ $h_9$ $h_{10}$ $h_{11}$ $h_{12}$ $h_{13}$ $h_{14}$ $h_{15}$
Indexing
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
$X_{idx}$ = [1,2,3,4,5,6]
$P_{idx}$ = [7,8]
$Y_{idx}$ = [9,10,11,12]

Figure 5: Infix-only architecture. Figure from (Li and Liang, 2021)

We decided to attach both prefix and infix vectors to our inputs simultaneously to aid in p*-tuning. We expected the additional parameters and augmented positional reasoning capacity of prompts to improve table-to-text summary performance, though also anticipated increased training time and worsened low-data behaviour during training. The low-data environment is largely a cause of the difference in size between training examples and prefix-tuning parameters - the E2E dataset only contains 50K examples, but is used to train a language model that has 500K trainable parameters even when prefix-tuning is applied. We expected potential overfitting which could be reduced by applying dropout during training.

In our study, we test the effect of applying both prefix and infix-tuning: [prefix; $x$; infix; $y$]. As shown in Figure 5, using prefix and infix in conjunction significantly underperformed prefix-only and infix-only environments (approximately and two to four times worse, compared to prefix-only and infix-only results).

**E2E (prefix + infix)**

| PL | LR | # Epochs | MET | R-L | CIDEr |
|---|---|---|---|---|---|
| 5 | $8 \cdot 10^{-5}$ | 5 | 15.2 | 42.5 | 0.59 |
| 15 | $1 \cdot 10^{-4}$ | 8 | 26.4 | 50.8 | 0.932 |

Table 5: Results from training prefix and infix, where LR is the learning rate, PL is the prefix length

### 3.4 Ablation 3: Modifying Prefix Initialization

We attempt to modify the prefix initialization, setting the prefix equal to “table2text” and observe the effect of introducing numerical characters. (Li and Liang, 2021) found that initialization to task-adjacent strings is helpful in low-data environments, such as the one experienced in tuning a large transformer-based autoregressive neural language model (eg. GPT-2) with the E2E dataset.

The results of this study are shown in Table 6.

**E2E (prefix initialization)**

| PI | BLEU | NIST | MET | R-L | CIDEr |
|---|---|---|---|---|---|
| ”table2text” | 69.8 | **8.82** | **48.9** | **76.1** | **2.64** |
| ”table-to-text:” | **70.3** | **8.82** | 46.3 | 72.1 | 2.46 |

Table 6: Results from training with novel prefix initialization “table2text”, where PI stands for the prefix initialization

An example of an input from the test set, and the respective gold output and model prediction are shown in Figure 6.

We notice that the model prediction captures the essence of the sample input - a table containing information on a restaurant review of Blue Spice. Using prefix-tuning, we are able to achieve a grammatical prediction that also contains all features described in the input table.

### 3.5 Ablation 4: Modify Decode Mode

The decode mode used for the baseline model was beam search. Beam search is proposed as a better alternative to exhaustive search and greedy decoding. Exhaustive search is computationally expensive since it involves calculating all possible sequences and greedy decoding is suboptimal since it only considers the single most probable token at each step. Instead, beam search reduces the scope of consideration to the top k most likely sequences where k refers to the size of the candidate window,

Sample input (test set):

name : Blue Spice | Type : pub | food : Chinese | area : city centre | family friendly : no | near : Rainbow Vegetarian Café < |endoftext| >

Gold output:

(1) Blue Spice, located near Rainbow Vegetarian Café in the city centre, is a pub that also sells Chinese food. Children should not visit.

(2) A pub named Blue Spice is located in the city centre. It is for adults and is close to Rainbow Vegetarian Café. They offer Chinese food.

(3) The Blue Spice is a pub that also serves Chinese food, it's located in the city center. The pub isn't very family friendly, but the proximity to the Rainbow Vegetarian Café makes up for it.

(4) Near Rainbow Vegetarian Café Blue Spice pub Chinese in city centre family friendly no. There is a pub Blue Spice located in the centre of the city that provides Chinese food. It is not family friendly and is located near Rainbow Vegetarian Café. located in the city center. right next to the Rainbow Vegetarian Café, the pub, Blue Spice, has wonderful Chinese food. The Blue Spice is perfect for date night, but i wouldn't recommend bringing your family there.

(5) Blue Spice is a pub located in the city centre. It has Chinese food and is not family friendly and is located near Rainbow Vegetarian Café.

Model prediction:

(1) Blue Spice is a pub that provides Chinese food It is located in the city centre. It is near Rainbow Vegetarian Café.

Figure 6: Sample input/gold/prediction tuple in the test set, evaluated on the prefix initialization ablation

Gold output:

(1) A coffee shop in the city centre area called Blue Spice.

(2) There is a coffee shop named Cocum located near Burger King. This coffee shop has a high customer rating.

Model output:

(1) Blue Spice is a coffee shop in the city centre.

(2) Cocum is a coffee shop located near Burger King. It has a high customer rating.

Figure 7: Results from the nucleus sampling ablation study

or beam. It presents an efficiency improvement over exhaustive search by pruning the paths that are not part of the top k.

An alternative decoding approach that is commonly used in lieu of beam search is nucleus sampling. Nucleus sampling is similar to Top-K sampling, which eliminates less probable words from consideration by focusing on only the top k most probable tokens. Nucleus sampling addresses a limitation of Top-K sampling, namely that the value of k needs to be determined beforehand. Instead, nucleus sampling is more flexible/dynamic: it considers the minimum set of words whose sum crosses a certain probability threshold and eliminates all other tokens from consideration. This approach allows the model to have the flexibility between focusing on a small set of tokens for which it has high confidence or enlarging the candidate set when there is higher uncertainty. The nucleus sampling formula is

$$\sum_{x \in V^{(p)}} P(x|x_{1:i-1}) \geq p. \tag{1}$$

In this study, we performed nucleus sampling and returned five results for text generation and selected the first result as the final output. The results are shown in 7; evidently, nucleus sampling produces comparable results to beam search but does not present an improvement.

**E2E (nucleus-sampling)**

|  | MET | R-L | CIDEr |
|---|---|---|---|
| Ours | 41.7 | 63.5 | 1.86 |

Table 7: Results from the nucleus sampling ablation study

We also include sample output of our model in comparison with the gold output in Figure 7. The results show that our model is able to capture much of the meaning presented in the target output although it expresses it with different words.

## 4 Discussion

In summary, we first replicated the baseline prefix-tuning model in the (Li and Liang, 2021) paper. This ended up occupying the majority of the research period due to the high ramp up required to understand the codebase and the code execution. The originally downloaded codebase from the paper's github repository was incomplete (missing evaluation data) and we eventually found the paper's Codalab repository, which we used for the remainder of the project steps.

After replicating the baseline results, we then augmented the model with four ablation studies. In the first, we tried six different prefix lengths. Results remained largely similar but we achieved slight improvement over the paper's baseline results with prefix length 20 (according to MET and CIDEr metrics). Throughout our paper, MET stands for METEOR anad R-L stands for the ROUGE-L metric.

In our first ablation, we sought to explore how prefix length affects accuracy and overfitting in a *closed-domain* dataset (E2E) that is smaller than the open-domain DART. We found that there is a threshold prefix length after which accuracy declines (15 or 20), so this may suggest that in smaller datasets, overfitting due to large prefix lengths is still a concern, because there is a higher chance that the distributions between the training and testing set are different.

In the second ablation study, we added an infix in addition to the prefix. This led to worse performance. We surmised that this could be because we were running the same number of epochs with now twice the number of optimizable parameters. To test this theory, we re-ran the mode with more epochs (8 instead of 5), and we notice in the second row of Table 5 that the accuracies increase across METEOR, ROUGE-L, and CIDEr with just 3 more epochs and a slightly larger learning rate. This may indicate that tuning prefix and infix in conjunction may need more epochs to achieve comparable results. This is an area for future research to delve into.

In the third ablation study, we initialized the prefix to "table2text", following the intuition that task-specific instructions such as "summarize this table" provide relevant context to natural language models. Among our two baselines and four ablations, prefix initialization achieved the highest accuracy across METEOR, ROUGE-L, and CIDEr metrics (Table 8). Figure 8 shows the prefix initializations the original authors tried. We find that our proposed prefix initialization of "table2text" outperforms all previous initializations and shows that even a numerical representation of "to" in "table-to-text" is able to be incorporated by the model.

In the final ablation study, we attempted to replace the existing decoding method, beam search, with nucleus sampling. Our results validated the author's selection of the beam search approach as nucleus sampling did not yield improved results.

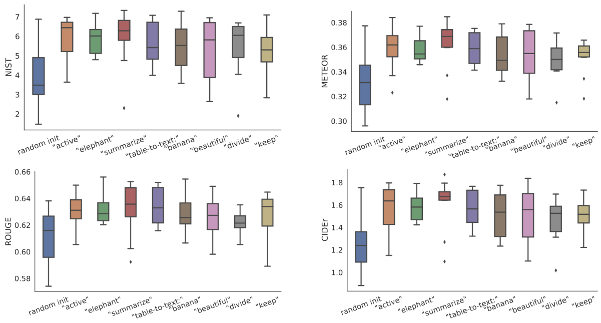

Figure 8: Prefix initializations that the original authors tried.(Li and Liang, 2021)

Table 8 shows our final results compared with the state of the art defined by (Li and Liang, 2021).

**E2E (prefix)**

| | MET | R-L | CIDEr |
|---|---|---|---|
| SOTA | 46.3 | 72.1 | 2.47 |
| Prefix-Only | 46.5 | 71.2 | 2.52 |
| Infix-Only | 46.0 | 70.5 | 2.41 |
| Prefix-Length (20) | 46.1 | 71.4 | 2.47 |
| Prefix + Infix | 26.4 | 50.8 | 0.932 |
| Nucleus Sampling | 41.7 | 63.5 | 1.86 |
| Prefix Initialization | **48.9** | **76.1** | **2.64** |

Table 8: Results from our ablation studies evaluated on test set compared to state of the art performance (SOTA is taken to be the results given by (Li and Liang, 2021))

## 5 Conclusion and Future Work

In conclusion, we were successful in reproducing the original author's table-to-text generation baseline of GPT-2 on the E2E dataset, and even exceeded their accuracy using the MET and CIDEr metrics (see Figure 1). In addition, we proposed and implemented four new ablations.

As for future work, we would like to explore alternatives to the GPT-2 baseline model. For example, we considered replacing it with XLNet by Carnegie Mellon University. Proposed as an alternative to BERT, XLNet is an autoregressive pretraining method that is able to capture bidirectional context. Initial attempts to substitute the underlying model were unsuccessful due to incompatabile configuration differences between the two models that required more time to debug and investigate.

## 6 Code and Reproducibility

In order to maximize reproducability, the code in this paper is available at https://github.com/RichardZhu123/cos484final. A Codalab from (Li and Liang, 2021)—which contains additional data and the libraries used in our implementation—is also available at https://worksheets.codalab.org/worksheets/0x16e0c8e7ab1f4b22aaccddc8b586541f

## Acknowledgments

We would like to acknowledge Professor Karthik Narasimhan, Mengzhou Xia, Ameet Deshpande, and Lisa Li for all of their help with the project. We're also incredibly grateful to both Percy Liang and Lisa for the insightful paper.

## References

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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. *CoRR*, abs/2005.14165.

Guanzheng Chen, Fangyu Liu, Zaiqiao Meng, and Shangsong Liang. 2022. Revisiting parameter-efficient tuning: Are we really there yet? *arXiv preprint arXiv:2202.07962*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The WebNLG challenge: Generating text from RDF data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, et al. 2021. Dart: Open-domain structured data record to text generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017a. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206.

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2017b. The E2E dataset: New challenges for end-to-end generation. *CoRR*, abs/1706.09254.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Sheng Shen, Daniel Fried, Jacob Andreas, and Dan Klein. 2019. Pragmatically informative text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4060–4067, Minneapolis, Minnesota. Association for Computational Linguistics.