Semiparametric Token-Sequence Co-Supervision

Anonymous ACL submission

Abstract

In this work, we introduce a semiparametric token-sequence co-supervision training method. It trains a language model by simultaneously leveraging supervision from the traditional next token prediction loss which is calculated over the parametric token embedding space and the next sequence prediction loss which is calculated over the nonparametric sequence embedding space. The nonparametric sequence embedding space is constructed by a separate language model tasked to condense an input text into a single representative embedding. Our experiments demonstrate that a model trained via both supervisions consistently surpasses models trained via each supervision independently. Analysis suggests that this co-supervision encourages a broader generalization capability across the model. Especially, the robustness of parametric token space which is established during the pretraining step tends to effectively enhance the stability of nonparametric sequence embedding space, a new space established by another language model. We will publicly release our model and code in URL.

1 Introduction

Language models are typically trained through next-token prediction (NTP), where the model forecasts the distribution of the next token's embedding, given a current token embedding (Touvron et al., 2023a; Brown et al., 2020; Zhang et al., 2022). This process relies on a language model head, which includes embeddings for the entire vocabulary. While this approach has demonstrated high performance, its reliance on predicting over a finite parametric vocabulary space restricts the models' expressivity (Min et al., 2022; Yang et al., 2017; Pappas et al., 2020). Also, such supervision constrains the model's predictive capabilities to only the next token, whereas humans can anticipate sequences of varying granularities highlighting a significant gap.

In this work, we aim to enhance the capabilities of language models by superposing parametric token embedding space and nonparametric sequence embedding space at the output space of a language model. Drawing on previous research that highlights the adaptable nature of language models' parametric token embedding space, we theorize that the language model can integrate a new embedding space alongside the model's existing parametric space and can also leverage the stable foundation of its parametric space established during the pretraining phase when integrating the new embedding space.

To this end, we introduce semiparametric token-sequence co-supervision, a novel training approach that trains a language model (Gen) by incorporating supervision from *both* the traditional next token prediction (NTP), calculated over the parametric token embedding space, and next sequence prediction (NSP), which is calculated over nonparametric sequence embedding space as in Figure 1. This nonparametric sequence embedding space is constructed by another language model, $\texttt{Emb}_{seq}$, which compresses the entire input text into a singular, representative embedding. The supervision is calculated via contrastive learning over embedding from the nonparametric embedding space and the output distribution from Gen.

We experiment across 10 information-seeking datasets in KILT and ALCE benchmarks. We compare a model trained via semiparametric token-sequence co-supervision, multi-task training over NTP and NSP, against a model trained on each supervision individually. Models trained with co-supervision consistently outperform models trained on separate supervisions by an average of 14.2, demonstrating that constructing a common space through co-supervision fosters the generalization and robustness of the language model. The nonparametric space under semiparametric token-sequence co-supervision is more stable com-

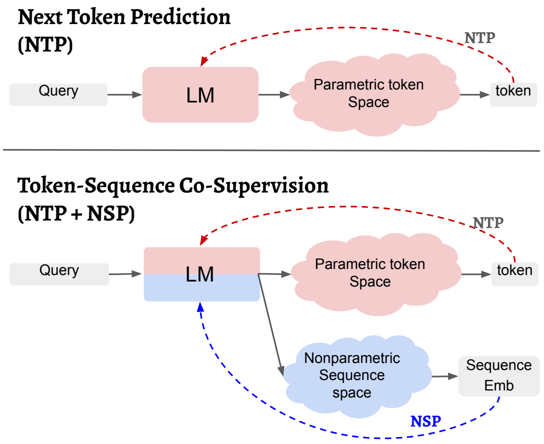

Figure 1: While previous methods train language models with next token prediction loss (NTP), semiparametric token-sequence co-supervision trains a language model in a *multi-task* manner where supervision from parametric token embedding space (NTP) and supervision from nonparametric sequence embedding space (NSP) flow simultaneously.

pared to models trained solely on NSP, suggesting that the robustness of the parametric space, established through pretraining, provides a solid foundation that enhances the stability of the nonparametric space. Also, unlike models trained only via NTP, the model trained via semiparametric token-sequence co-supervision tends to effectively use knowledge from the nonparametric space during generation, suggesting a shift from rote learning to active knowledge utilization.

Models trained through token-sequence co-supervision demonstrate a notable enhancement over models trained with each type of supervision independently, especially on out-of-domain datasets, with an average improvement rate of 6.6 compared to in-domain datasets. Also, the token-sequence co-supervision promotes robust interaction between the parametric token space and the nonparametric sequence space; Gen tends to generate responses by utilizing the knowledge from the nonparametric space. Moreover, the inherent distribution of $\texttt{Emb}_{seq}$, a pretrained language model (LM), influences the overall performance where aligning $\texttt{Emb}_{seq}$ and Gen to the same pretrained LM thereby the same distribution, contributes to a more stable training process.

2 Related Works

Aligning two different models Various studies have explored ways to align two different models. In multi-modal tasks, efforts have been made to connect pretrained vision-only and language-only models, either by employing cross-attention mechanisms (Alayrac et al., 2022) or by aligning the vision encoder's output embedding with the language model's input space (Liu et al., 2023). Also, aligning a language model with another multilingual language model enables the model to perform multilingual tasks by leveraging the distribution from the multilingual model (Bansal et al., 2024; Yoon et al., 2024). Moreover, recent research has focused on retrieval-augmented language models that align a retrieval model with a language model to mitigate the issue of hallucination in language models. Studies suggest that aligning these two models leads to improved performance compared to training them separately (Lin et al., 2023; Shi et al., 2023). semiparametric token-sequence co-supervision also aims to train on aligning two different models but is unique in that it focuses on aligning the two models at the output space of the language model rather than the input space.

Language Models with Nonparametric Embeddings Integrating nonparametric embeddings into language models has consistently demonstrated advantages. This approach enhances the expressiveness beyond the inherent capabilities of language models (Khandelwal et al., 2019; Zhong et al., 2022). Also, leveraging the rich contextual knowledge encapsulated within nonparametric embeddings effectively reduces instances of hallucination and improves the generation of accurate and factual content (Lewis et al., 2020; Borgeaud et al., 2022; Guu et al., 2020). Moreover, it show high performance for rare and unseen cases as such tend to not exist in model paramteric space (Lee et al., 2023b; Min et al., 2022). semiparametric token-sequence co-supervision also leverages the nonparametric embedding space but is unique in that it *trains* the model to utilize *both* the parametric and nonparametric embedding space, where the nonparametric embedding space is not static at the training step, but is trainable making the nonparametric space more adaptable to the well-constructed parametric embedding space.

3 Semiparametric Token-Sequence Co-Supervision

In Section 3.1, we delve into our interpretation of the Next Token Prediction (NTP), first laying the foundation for our hypothesis. Subsequently, in Section 3.2, we explore next sequence prediction (NSP), an extension of next token prediction to

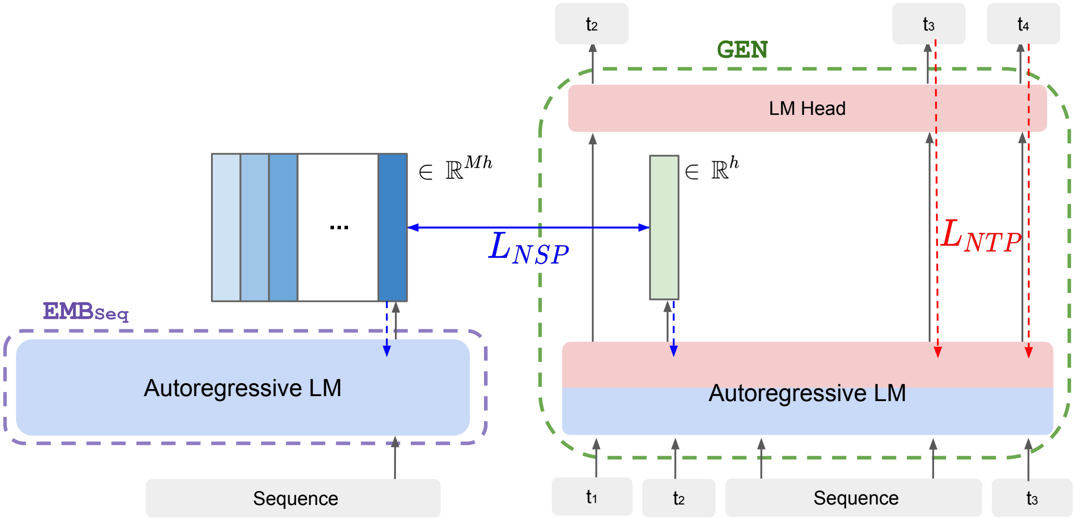

Figure 2: Overview of Semiparametric token-sequence co-supervision. Gen is an autoregressive LM with LM head on top, which is trained with co-supervision over parametric token embedding space (L_{NTP}) and nonparametric sequence embedding space (L_{NSP}). Emb_{seq}, another autoregressive LM constructs nonparametric sequence embedding space with the output embeddings when given sequence as input. t_i indicates tokens, h indicates dimension size of hidden state, and M indicates number of sequences in a batch(Refer Appendix A.1 for datailed calculation).

nonparametric sequence embedding space. In Section 3.3, to test our hypothesis we introduce Semiparametric token-sequence co-supervision, a training method with supervision from both parametric token embedding space (NTP) and nonparametric sequence embedding space (NSP) in a simultaneous manner.

3.1 Revisiting Next Token Prediction

We revisit the conventional approach of Next Token Prediction (NTP), which forms the foundation of most modern language models (LMs). NTP is a process to predict token t over the vocabulary set V when given the preceding tokens $t_1, \ldots, t_k$ to a language model:

$$\text{argmax}_{t \in V} \; P(t|t_1, \ldots, t_k) \tag{1}$$

As shown in Figure 2, we specify Gen as an autoregressive language model, consisting of a language model (LM) and language model head (LM Head). LM Head $\mathbf{W}_V$ ($\in \mathbb{R}^{|V| \times h}$) is a linear layer where h denotes the model hidden state dimension. When given sequence of tokens $t_1, \ldots, t_k$ to LM, it returns a hidden state vector $\mathbf{q}_k$ ($\in \mathbb{R}^h$). The hidden state is calculated with LM Head which returns the probability distribution over vocabulary size. Thereby Equation 1 can be reformulated as:

$$\text{argmax}_{t \in V} \mathbf{W}_V \mathbf{q}_k \tag{2}$$

Equation 2 can be interpreted as a retrieving stage, indicating that the parametric token embedding space $\mathbf{W}_V$ (LM Head) which consists of model vocab set determines the corpus (the range of objects from which "what" we will retrieve) when given $\mathbf{q}_k$, the hidden state embedding as a query. As the conventional language modeling paradigm only provides supervision over a fixed vocabulary-sized token embedding space $\mathbf{W}_V$, the usage was limited to predicting the most probable next token embedding. However, with such interpretation, when given multiple supervision from various embedding spaces, the methodology is extendable to predicting not only token embedding $\mathbf{W}_V$ but also various representatives in any other non-parametric spaces.

3.2 Next Sequence Prediction

Broadening the scope of next token prediction (NTP), we explore the domain of sequence-level embedding space $\mathbf{W}_\mathcal{C}$. In natural language processing, many tasks extend beyond merely predicting the next token; they necessitate the utilization of sequence-level knowledge such as contexts from external memory from a corpus (Zhong et al., 2022; Hao et al., 2023; Lewis et al., 2020). This is where the next sequence prediction (NSP) becomes invaluable. Unlike traditional NTP, which operates over parametric token embedding space, NSP interacts with nonparametric sequence embedding space. NSP allows models to anticipate and generate answers based on given *sequences* on-demand, mirroring the human-like ability to anticipate sequences of varying granularities and its ability to refer to external sequences while answering a question.

When equipped with an embedding space $\mathbf{W}_{\mathcal{C}}$ representing embedding space constructed with a corpus set of sequences $\mathcal{C}$, NSP is a process to predict the sequence s akin to Equation 2.

$$\text{argmax}_{s \in \mathcal{C}} \mathbf{W}_{\mathcal{C}} \mathbf{q}_k$$

Specifically, the embedding space $\mathbf{W}_{\mathcal{C}}$ is constructed by the last hidden state of another autoregressive model $\texttt{Emb}_{seq}$; $\mathbf{W}_{\mathcal{C}}$ is a nonparametric embedding space constructed with a set of last hidden state embedding $\mathbf{s}$ where $\mathbf{s} = \texttt{Emb}_{seq}(s)$ for $s \in \mathcal{C}$.

The concept of augmenting next token prediction (NTP) supervision with additional guidance has been previously explored. However, the motivations are distinct from those behind semiparametric token-sequence co-supervision. Our objective is to leverage co-supervision to foster a unified space that bridges the nonparametric sequence embedding space with the parametric token embedding space. This approach emphasizes supervision in identifying the appropriate relevant sequence from a corpus within the nonparametric space. In contrast, earlier efforts, such as the next sentence prediction (NSP) feature in BERT (Devlin et al., 2018), focused on a simpler binary supervision task that determines the relevance of a succeeding sentence. Also to the best of our knowledge, semiparametric token-sequence co-supervision is a first approach of training autoregressive language model with co-supervision apart from only NTP.

3.3 Co-Supervision

To test the hypothesis, we introduce Semiparametric token-sequence co-supervision as shown in Figure 2. It trains a single autoregressive language model Gen by incorporating supervision from both the traditional next token prediction (NTP) which is calculated over the parametric token embedding space and next sequence prediction (NSP), which is calculated over nonparametric sequence embedding space.

For NTP, we apply an ordinary casual language modeling loss function. When given input X:

$$L_{\text{NTP}} = -\frac{1}{|X|} \sum_{t_i \in X} log P_{\texttt{Gen}}(t_i | t_{<i}) \tag{3}$$

For NSP, we apply the contrastive InfoNCE loss (Karpukhin et al., 2020; Izacard et al., 2021); when given a query embedding, positive sequence pair relevant to the query, and a pool of negative sequences unrelated to the query, $\texttt{Emb}_{seq}$ and Gen are trained to maximize the similarity between the query embedding and the positive pair and minimize the similarity with the negatives pairs. As it is impractical to dump all embeddings of a corpus set every step, here we approximate softmax over all corpus by softmax over positive and negative pairs in the same batch (Karpukhin et al., 2020; Izacard et al., 2021). We consider other sequences in the same batch as negatives (in-batch negatives). Formally, given a query embedding $\mathbf{q}$, positive pair passage embedding $\mathbf{c}_i^+$, and negative pairs $\mathbf{c}_1^-, \cdots, \mathbf{c}_{M-1}^-$ where M is the number of sequences in a batch, NSP is calculated via:

$$L_{\text{NSP}}(\mathbf{q}_i, \mathbf{c}_i^+, \mathbf{c}_1^-, \ldots, \mathbf{c}_{M-1}^-)$$
$$= -\log \frac{e^{\text{sim}(\mathbf{q}_i, \mathbf{c}_i^+)}}{e^{\text{sim}(\mathbf{q}_i, \mathbf{c}_i^+)} + \sum_{j=1}^{M-1} e^{\text{sim}(\mathbf{q}_i, \mathbf{c}_j^-)}} \tag{4}$$

Thereby total loss over semiparametric token-sequence co-supervision is calculated as:

$$L_{\text{co-supervision}} = L_{\text{NTP}} + \lambda L_{\text{NSP}} \tag{5}$$

where λ is the weight parameter to match the loss scale between L_{NSP} and L_{NTP} as it flows through Gen together.

4 Implementation Details

In this section, we share implementation details of experiments over information-seeking datasets. In Section 4.1, we share details of the problem setup. In Section 4.2 and Section 4.3, we describe details of how we train a language model (Gen) with both supervision of next token prediction (NTP) and next sequence prediction (NSP) simultaneously, the inference step of the trained model, respectively.

4.1 Problem Setup

For details, we start with an explanation of notations and training instances. Gen is the trainable language model that trains over by co-supervision from both NTP and NSP. $\texttt{Emb}_{seq}$ is the trainable language model that constructs the nonparametric sequence embedding space and which calculates over Gen output embedding for NSP loss.

Training instances are in a form of "$q, n_1, [\text{NSP}], c_1, n_2, \cdots n_i, [\text{NSP}], c_i, \cdots$" (Asai et al., 2023) (Examples in Appendix B.2). q is an input query. [NSP] is a special token indicating the model to calculate over sequence embedding space, in other words when the model itself considers external knowledge is necessary. c_i is a relevant sequence when given sequences before [NSP] as input. For example, c_1 is the relevant sequence when given q, n_1 as an input. The number of [NSP] varies from 0 to multiple differing by how many times the query necessitates retrieval (external knowledge) during generation.

4.2 Training

Figure 2 shows the overview of how we train a language model Gen with semiparametric token-sequence co-supervision, where both L_{NTP} (equation 3) and L_{NSP} (equation 4) flows through a language model Gen simultaneously. For L_{NSP} loss, we train another language model $\texttt{Emb}_{seq}$ together which constructs a nonparametric sequence embedding space. Specifically, the query embedding ($\mathbf{q}$) and sequence embeddings ($\mathbf{c}$) to calculate L_{NSP} are calculated by:

$$\mathbf{q} = \texttt{Gen}\left(\text{query[NSP]}\right)[-1]$$
$$\mathbf{c} = \texttt{Emb}_{seq}\left(\text{<s>context</s>}\right)[-1]$$

which is the last layer token representation of [NSP] from Gen and last layer token representation of end-of-sequence token from $\texttt{Emb}_{seq}$, respectively. Thereby the gradient of L_{NSP} flows through both $\texttt{Emb}_{seq}$ and Gen. Whereas for L_{NTP}, the gradient only flows through Gen.

4.3 Inference

During the inference step, we first dump all context embedding with $\texttt{Emb}_{seq}$ during the offline time. Given a set of sequences, we feed each sequence into $\texttt{Emb}_{seq}$ and extract representative embedding $\mathbf{c}$.After the dumping, we get a context embedding matrix of $\mathbf{C} \in \mathbb{R}^{h \times M}$.

After dumping, in the online step, $\texttt{Emb}_{seq}$ is no longer necessary and we only need Gen. The generated response is in the same form as the training instances "$q, n_1,$ [NSP]$, c_1, n_2, \cdots n_i,$ [NSP]$, c_i, \cdots$" (Section 4.1).

When Gen generates [NSP], it treats the last layer representation of [NSP], generated after inputting sequences up to [NSP] $(q, \cdots, n_i)$, as the query embedding $\mathbf{q}$. This embedding $\mathbf{q}$ is then calculated over the context embedding matrix $\mathbf{C}$ to find the most relevant sequence (c_i) for the query. c_i is added after [NSP], allowing the generation process to proceed based on the sequence. When Gen generates a token other than [NSP], it functions the same as a standard language model, selecting the next token based on the highest probability through the language modeling head. See Appendix A.2 for more details.

5 Experiments Setup

Baseline For analysis of how co-supervision affects model performance, we train a baseline model the same as in Section 4.2 but with each supervision separately; NTP is calculated in the same way whereas NSP is calculated between output embeddings of $\texttt{Emb}_{seq}$ ($\mathbf{q}$ and $\mathbf{c}$).

$$\mathbf{q} = \texttt{Emb}_{seq}\left(\text{query[NSP]}\right)[-1]$$
$$\mathbf{c} = \texttt{Emb}_{seq}\left(\text{<s>context</s>}\right)[-1]$$

Metric To measure how training a language model with both supervision of NTP and NSP shows different aspects over the model trained with each supervision separately, we measure model performance via three metrics. Correctness and grounding performance measures how well the parametric token space is built and how well the two spaces interact. Retrieval performance mainly measures how well the nonparametric sequence space is built. *Correctness (Cor)* evaluates how well the model generates a response thereby answering the given query for each task. For instance, in the case of question answering (QA), correctness is measured by answer accuracy (Table 5 in the Appendix). *Retrieval* performance measures whether the model finds relevant paragraphs to answer the given question which requires well-constructed nonparametric sequence space. *Grounding (Gr)* performance evaluates how well the model generates based on given external knowledge. Following the approach of previous works (Gao et al., 2023; Lee et al., 2023a), we use TRUE (Honovich et al., 2022), a T5-11B model finetuned on various NLI datasets, to see whether the external knowledge entails a part of the response that is generated based on the knowledge. More details of the metrics are in Appendix B.1.

Training Dataset All models undergo training using the dataset provided by Asai et al. (2023), featuring a diverse range of instruction-following datasets. The dataset contains information-seeking questions paired with long-form responses, where a set of sequences are annotated within the responses (example form in Section 4.1). As the dataset includes instances beyond our scope such as cases where the matching sequence is irrelevant to the response as Asai et al. (2023) aims to train model self-critique, a filtering process was applied. This refined the dataset to 42,932 instances suitable for our research objectives. Detailed information about the dataset and our filtering approach can be found in the Appendix B.2.

Evaluation Dataset We conduct evaluations on two benchmarks: KILT (Petroni et al., 2021) and ALCE (Gao et al., 2023). Some of these datasets overlap with the training dataset, categorized as in-domain datasets, while others are considered out-of-domain datasets. While the ALCE setting shows closer alignment of our training dataset, as the benchmark does not contain annotation of relevant sequences, we adapt the KILT benchmark to conform to the ALCE setting. For this reason, we mainly focus analysis on the KILT benchmark, which we can measure all three metrics. Further details on the evaluation datasets are available in Appendix B.3.

Training details We use pretrained Llama2 7B (Touvron et al., 2023b) as an initial model for both `Gen` and $\texttt{Emb}_{seq}$. We use 8 Nvidia A100 for the experiments. We also set the base hyperparameter as epoch 3, batch size 8, learning rate of 2e-5 and a decayed rate gamma of 0.85 every 1 epoch, and AdamW optimizer (Loshchilov and Hutter, 2019) with no decay across all the experiments. For experimenting semiparametric token-sequence co-supervision, we set the weight λ as 0.01 while training. We also apply gradient clipping to the `Gen` model, with a maximum norm equal to 1.

6 Experimental Results & Analysis

Table 1 shows the overall performance on the KILT benchmark and Table 2 shows the overall performance on the ALCE benchmark when given 20 sequences (Top 20) or 100 sequences (Top 100) as a corpus. Both tables show that models trained with semiparametric token-sequence co-supervision consistently outperform models trained under each type of supervision separately. This suggests that this co-supervision encourages a broader generalization capability throughout the model. In this section, we delve into the impact of each type of supervision on the model's performance and explore the effects of co-supervision. **From now, for simplicity we name the model trained with semiparametric token-sequence co-supervision as NTP + NSP and the model trained under each supervision separately as NTP[1].**

Nonparametric sequence embedding space under semiparametric token-sequence co-supervision is more stable Training a language model with both supervisions from L_{NTP} and L_{NSP} consistently shows higher retrieval performance (average of +16.6) over those trained only with L_{NSP}. Especially, the performance gap tends to increase as corpus size increases and over out-of-domain (improvement rate of 35.04 for in-domain and 41.67 for out-of-domain). Such results indicate that the nonparametric sequence embedding space constructed through $\texttt{Emb}_{seq}$ is more stable when trained together with supervision from parametric token embedding space. As previous research has shown that new embeddings (tokens) can easily adapt to well-established parametric token embedding space of the model (Hao et al., 2023; Schick et al., 2023), we hypothesize such adaptability applies to nonparametric sequence embedding space; the robustness of the parametric token embedding space, established during the pretraining step provides a solid foundation that enhances the stability of the nonparametric space.

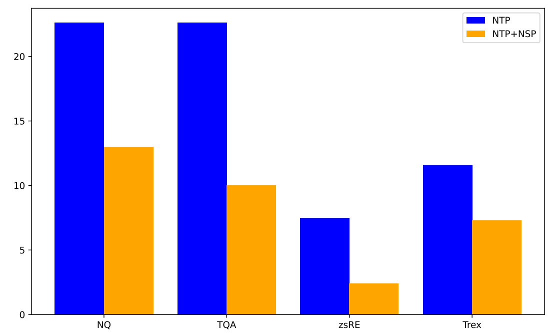

Figure 3: Reduction rate of correctness when considering those correct by parametric knowledge as wrong.

Co-supervision encourages high interaction between the two spaces Figure 3 (Numbers are in Table 7 of Appendix) shows the degradation rate of correctness when removing the ones that are correct by the parametric knowledge (categoriz-

[1] Please note that NTP is not only trained with NTP but also NSP but separately. We name it NTP for simplicity

	Retriever	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr
			NQ*			WoW*			FEVER*			ELI5	
Top 20	NTP	62.0	49.7	51.5	31.0	14.7	44.9	58.0	57.1	5.8	30.9	**21.8**	**9.3**
	NTP + NSP	**65.1**	**55.7**	**62.6**	**49.8**	**15.7**	**63.7**	**77.5**	**65.2**	**28.0**	**36.3**	21.5	8.7
Top 100	NTP	35.7	38.1	43.0	14.7	13.4	40.0	28.8	56.7	5.2	12.7	**21.9**	7.0
	NTP + NSP	**56.8**	**50.5**	**58.7**	**36.9**	**14.8**	**61.3**	**66.2**	**64.3**	**26.1**	**21.0**	21.6	**10.2**
			zsRE			T-REx			TriviaQA			HotpotQA	
Top 20	NTP	51.2	40.3	54.6	60.6	40.6	48.5	63.0	65.2	41.7	30.2	29.9	43.1
	NTP + NSP	**80.5**	**59.6**	**74.0**	**75.5**	**67.1**	**63.9**	**74.5**	**71.7**	**47.9**	**55.6**	**37.9**	**48.9**
Top 100	NTP	32.8	27.1	47.2	47.4	30.8	45.1	46.5	54.9	37.7	12.6	19.6	11.8
	NTP + NSP	**71.2**	**53.4**	**70.0**	**67.7**	**58.9**	**59.1**	**67.3**	**68.0**	**45.7**	**46.2**	**34.3**	**45.5**

Table 1: Overall performance of NTP + NSP (model trained with semiparametric token-sequence co-supervision) and NTP (models trained under each type of supervision separately) in KILT benchmark. Datasets with * on top indicate in-domain datasets.

		ASQA*		ELI5	
		Cor	Gr	Cor	Gr
Top 5	NTP	28.4	42.5	**9.7**	8.9
	NTP + NSP	**31.8**	**44.0**	9.3	**10.3**
Top 100	NTP	20.2	31.1	9.9	13.1
	NTP + NSP	**26.3**	**36.8**	**10.5**	**13.4**

Table 2: Overall performance of NTP + NSP and NTP in ALCE benchmarks. Dataset with * on top indicates in-domain datasets.

ing responses associated with incorrect paragraphs as wrong). Models trained with semiparametric token-sequence co-supervision show less degradation compared to the one with separate supervision, highlighting that co-supervising encourages interaction between token and sequence embedding space as it is trained to share a common space. Also, it suggests that the model leverages the contextual knowledge within the nonparametric space for generating responses, as supported by previous research (Min et al., 2022; Lee et al., 2023b), rather than depending exclusively on its parametric knowledge base. Even when limiting Gen to only parametric space during inference, thus preventing it from accessing nonparametric sequence embedding space, the model trained via co-supervision exhibits a more significant drop in correctness (from 45.7 to 32.8) compared to the model only with supervision from NTP (from 32.8 to 32.0), indicating semiparametric token-sequence co-supervision encourages on using the knowledge from nonparametric sequence space rather than mere memorization, utilizing knowledge from its own parametric space.

Co-supervision enhances general understanding over varying input distribution. Models trained with semiparametric token-sequence co-supervision consistently outperform those trained under indivico-supervision in terms of correctness and grounding performance across both KILT and ALCE benchmarks. This advantage is more pronounced with larger corpus sizes, with Top 100 showing more significant improvements than Top 20. The performance gap is particularly noticeable in datasets such as FEVER, T-REx, and zsRE, largely due to differences in generalization capabilities. Models under co-supervision demonstrate a broader understanding across diverse input distributions, whereas models trained solely with next token prediction (NTP) struggle with unique input formats. For example, T-REx and zsRE, which are slot-filling tasks, present unique formats ("subject [SEP] relationship type") that pose challenges. Additionally, in FEVER, the issue of low grounding performance arises from models generating simple answers without providing evidence, despite being prompted for evidence-based responses.

Flowing loss over the sequences when calculating L_{NTP} in co-supervision tends to make the model memorize the knowledge rather than utilizing the knowledge from nonparametric embedding space One interesting factor we found is that when we do not mask sequences when calculating L_{NTP} of token-sequence co-supervision Gen tend to rely more on their memorized parametric knowledge compared to models trained with sequences masked, which tends to utilize the external knowledge retrieved from the nonparametric sequence embedding space. Such a tendency aligns with the findings of Mallen et al. (2022), where the model tends to depend on retrieved knowledge more when it lacks familiarity with the information (long-tail knowledge). By calculating generation loss over the context, the model tends to encode the context knowledge in its parameters, leading to a reduced reliance on retrieved knowledge and more on its own knowledge.

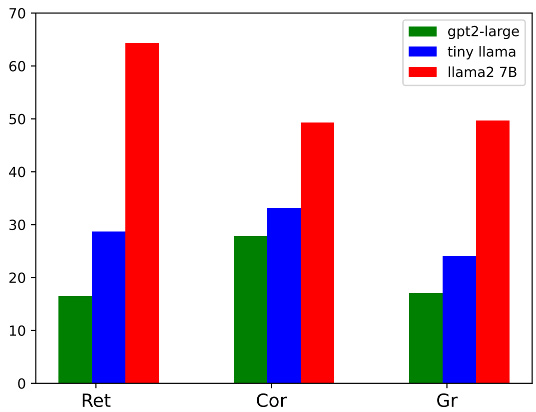

Figure 4: Overall performance of how different $\texttt{Emb}_{seq}$, which constructs the nonparametric sequence embedding space, affects the overall performance when training with NTP + NSP. We experiment over 3 different models, GPT2-large, TinyLlama, Llama2-7B.

How does the choice of $\texttt{Emb}_{seq}$ affect performance? In our experiments, we investigated how different versions of $\texttt{Emb}_{seq}$ impact overall performance, specifically focusing on how the output embeddings from $\texttt{Emb}_{seq}$ contribute to constructing the nonparametric sequence embedding space. The average results, illustrated in Figure 4, compare three distinct choices for $\texttt{Emb}_{seq}$: GPT2-large (Radford et al., 2019), TinyLlama (Zhang et al., 2024), and Llama2-7B. The findings suggest that the specific distribution inherent to each pretrained language model influences the performance of Gen. Notably, when Gen and $\texttt{Emb}_{seq}$ are derived from the same model—thus sharing the same distribution—the training process appears more stable. This observation underlines the significance of distribution compatibility between Gen and $\texttt{Emb}_{seq}$ in enhancing model training and performance.

Affect of weight lambda (λ) when training semiparametric token-sequence co-supervision We investigate how different weights ($\{10^{-1}$, 10^{-2}, $10^{-3}\}$) between the next token prediction supervision (L_{NTP}) and next sequence prediction supervision (L_{NSP}) affect the model's performance,

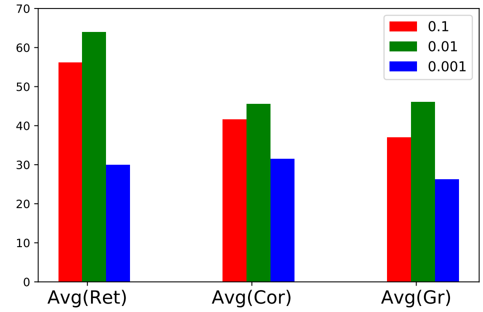

Figure 5: Average performance of each metric over 8 datasets in KILT when changing weight parameter λ of NTP + NSP.

which is the lambda weight in Equation 5. In our setup, as a single generation model Gen receives co-supervision from both the parametric token embedding space and the nonparametric sequence embedding space, the weight determines the balance of influence between these two spaces on the model's training. Figure 5 (numbers in Table 6 in the Appendix) illustrates that a weight of 10^{-3} results in poor retrieval performance, indicating challenges in grasping and stabilizing the nonparametric sequence embedding space. Moreover, at a weight of 10^{-1}, the model's generation ability tends to decline, suggesting that it rather ruins the well-formed parametric token embedding space. This analysis underscores the critical role of balancing supervision from both embedding spaces to optimize model performance across various metrics.

7 Conclusion

In this paper, we propose a semiparametric token-sequence co-supervision training method, which trains a single autoregressive language model with both supervision from parametric token embedding space and nonparametric sequence embedding space in a simultaneous manner. Experiments over 10 information-seeking datasets show that such co-supervision consistently outperforms models trained with each supervision separately of average +14.2, demonstrating that constructing a common space through co-supervision fosters the generalization and robustness of the language model. Such a method is not only limited to sequence space but can be expandable to any embedding space which we leave as future work.

8 Limitation

Due to resource constraints, our experimentation did not extend to altering Gen with other pretrained LMs such as Mistral (Jiang et al., 2023). Moreover, as our primary objective is not to develop the optimal model through token-sequence co-supervision but rather to explore the efficacy and implications of this approach, our hyperparameter tuning was focused on key factors such as the weight parameter λ, batch size, among others, rather than conducting an extensive hyperparameter search.

9 Ethics

While our model benefits from token-sequence co-supervision, which enhances its ability to utilize external knowledge, we must acknowledge potential ethical implications. Notably, we have not yet explored how the model behaves when generating sequences containing external knowledge that may contain toxic information. For instance, consider a scenario where the model generates text containing biased or harmful information sourced from external knowledge bases. It is essential to conduct further research and consider ethical implications to ensure that the model's outputs align with societal values and norms, such as refusing to predict from such text. Addressing these concerns will be crucial for responsibly deploying the model in real-world applications.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.

Rachit Bansal, Bidisha Samanta, Siddharth Dalmia, Nitish Gupta, Shikhar Vashishth, Sriram Ganapathy, Abhishek Bapna, Prateek Jain, and Partha Talukdar. 2024. Llm augmented llms: Expanding capabilities through composition. *arXiv preprint arXiv:2401.02412*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *ICLR*.

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Paslaru Bontas Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *LREC*.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. Eli5: Long form question answering. *ArXiv*.

Tianyu Gao, Howard Yen, Jiatong Yu, and Danqi Chen. 2023. Enabling large language models to generate text with citations.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR.

Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. 2023. Toolkengpt: Augmenting frozen language models with massive tools via tool embeddings. *ArXiv*, abs/2305.11554.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. True: Re-evaluating factual consistency evaluation. *arXiv preprint arXiv:2204.04991*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *TACL*.

Hyunji Lee, Sejune Joo, Chaeeun Kim, Joel Jang, Doyoung Kim, Kyoung-Woon On, and Minjoon Seo. 2023a. How well do large language models truly ground? *ArXiv*, abs/2311.09069.

Hyunji Lee, Jaeyoung Kim, Hoyeon Chang, Hanseok Oh, Sohee Yang, Vladimir Karpukhin, Yi Lu, and Minjoon Seo. 2023b. Nonparametric decoding for generative retrieval. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12642–12661.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *CoNLL*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kuttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401.

Xi Victoria Lin, Xilun Chen, Mingda Chen, Weijia Shi, Maria Lomeli, Rich James, Pedro Rodriguez, Jacob Kahn, Gergely Szilvasy, Mike Lewis, Luke Zettlemoyer, and Scott Yih. 2023. Ra-dit: Retrieval-augmented dual instruction tuning. *ArXiv*, abs/2310.01352.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. When not to trust language models: Investigating effectiveness and limitations of parametric and non-parametric memories. *ArXiv*, abs/2212.10511.

Sewon Min, Weijia Shi, Mike Lewis, Xilun Chen, Wen-tau Yih, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Nonparametric masked language modeling. *arXiv preprint arXiv:2212.01349*.

Nikolaos Pappas, Phoebe Mulcaire, and Noah A Smith. 2020. Grounded compositional outputs for adaptive language modeling. *arXiv preprint arXiv:2009.11523*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *ArXiv*, abs/2302.04761.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *NACCL*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023a. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael

Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. 2017. Breaking the softmax bottleneck: A high-rank rnn language model. *arXiv preprint arXiv:1711.03953*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Dongkeun Yoon, Joel Jang, Sungdong Kim, Seungone Kim, Sheikh Shafayat, and Minjoon Seo. 2024. Langbridge: Multilingual reasoning without multilingual supervision. *arXiv preprint arXiv:2401.10695*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Zexuan Zhong, Tao Lei, and Danqi Chen. 2022. Training language models with memory augmentation. *arXiv preprint arXiv:2205.12674*.

A NTP + NSP

A.1 Gathering in-batch negatives

Suppose we have B batched instances $D_1, \dots D_B$ for each training step, and each D_i has $c_{i,1}, \dots c_{i,l_i}$ reference context($l_i \geq 1$). We can collect $M = \sum_{i=1}^{B} l_i$ context embeddings(after counting duplicates) and the same amount of query embedding, each referring to one positive target. We set the target context embedding as the positive one, while setting the rest $M-1$ as the in-batch negative samples.

During the experiments, we gather all the context embeddings across all 8 GPUs to increase the number of in-batch negatives. We ensure at least 63 [2], but the exact number differs across training steps(experimentally about 80 to 90).

[2]at least 1 context per instance, 8 per each GPU, a total of 8 GPUs infer 64 total reference context embeddings, yielding at least $64 - 1 = 63$ in-batch negatives.

A.2 Inference

We run inference of our trained models using 1 A6000 GPU with 48GB memory.

A.3 Parsing

To distinguish where the model generates by grounding on the retrieved sequence and where the model generates in a freeform, we add two special tokens [CS] and [CE] which each indicate the start of the generation by grounding on the retrieved sequence and the end, respectively. In other words, when the model generates a response in the form of "$q, n_1, [\text{NSP}], c_1, g_1, \cdots n_i, [\text{NSP}], c_i, g_i, \cdots$", the part between [CS] and [CE] is the g_i part and the rest is n_i indicating freeform generation.

B Experiments Setup

B.1 Details of metrics

We assess the generation results in three axes: correctness, retrieval performance, and grounding performance

Correctness Correctness evaluates how well the model answers the given query for each task. For each dataset, we chose the metric to evaluate following the metric used in its official paper. Details for each dataset is in Table 7.

Retrieval Performance Retrieval performance measures whether the model retrieves relevant paragraphs to answer the given question. We measure in two aspects, whether the gold paragraph exists within retrieved paragraphs (Ret) and retrieval precision to assess how many of the model-retrieved paragraphs contain gold paragraphs (Ret-P). For example, when the model retrieves three different paragraphs {P1, P2, P3} while generating a response and only one of them {P2} is the gold paragraph, Ret will be 100 since there is a gold paragraph in the set of retrieved paragraphs while Ret-P is $\frac{1}{3}$ since only one is correct. Please note that we measure the metric by considering both gold and retrieved as a set; when the same paragraph is retrieved twice, we consider it as one during the calculation.

Grounding Performance Grounding performance evaluates how well the model generates based on given external knowledge. Following the approach of previous works (Gao et al., 2023; Lee et al., 2023a), we use TRUE (Honovich et al., 2022), a T5-11B model finetuned on various NLI

Source	Name	Instance Num
Open-Instruct	GPT-4 Alpaca	6363
	Stanford Alpaca	7826
	FLAN-V2	720
	ShareGPT	2544
	Open Assistant 1	2671
KILT	Wizard of Wikipedia	2159
	Natural Questions	12306
	FEVER	5502
HF Dataset	OpenBookQA	54
	Arc-Easy	183
ASQA	ASQA	2732

Table 3: Statistic and characteristics of training dataset (42932 instances). Instance Num inside the parentheses indicate the number of corresponding instances within all the filtered instances (43060 instances)

datasets, to see whether the external knowledge entails a part of the response that is generated based on the knowledge. To be more specific, as we add special tokens [CS] and [CE] to distinguish between natural form generation and generation based on given external knowledge (grounding generation); [CS] indicates the starting point of grounding generation, and [CE] indicates the endpoint. Thereby we parse the generated response into pairs (external knowledge, grounding generation) and calculate the average of whether the external knowledge entails the grounding generation in sentence wise of grounding generation.(Further details on parsing the output are available in Appendix A.2.)

B.2 Details of training dataset

Filtering We apply 3 filtering from Asai et al. (2023) to make our training dataset.

1. Relevance and Support: We retained instances with relevant and fully or partially supported citations to ensure dataset integrity.
2. Citation Count: Instances without citations or with more than 3 citations were filtered out to balance complexity and informativeness.
3. Length Constraint: We removed instances exceeding 1024 tokens (input + output) to optimize computational efficiency.

These criteria yielded a final training dataset of 42932 instances, poised to provide high-quality examples for effective model training.

Statistics Among 42.9k training instances, 32.1k instances contain 1 [NSP], 6.7k instances contain 2 [NSP], and 4k instances contain 3 [NSP] in the output. The source of these instances is shown in Table 3.

Single [NSP] Example Question: "which river has the largest basin in world"

Answer: "[NSP] Amazon River and Crenarchaeota. The Amazon has over 1,100 tributaries, 12 of which are over long. Some of the more notable ones are: Amazon River The Amazon River (, ; Spanish and) in South America is the largest river by discharge volume of water in the world, and by some definitions it is the longest. The headwaters of the Apurímac River on Nevado Mismi had been considered for nearly a century as the Amazon's most distant source, until a 2014 study found it to be the Cordillera Rumi Cruz at the headwaters of the Mantaro River in Peru. The Mantaro and Question: which river has the largest basin in world [CS] Amazon [CE] "

Multi [NSP] Example Question: "Why do corporations lobby for laws that impoverish lower class citizens?"

Answer: "[NSP] Corporate capitalism power and influence corporations and large business interest groups have over government policy, including the policies of regulatory agencies and influencing political campaigns (see corporate welfare). Many social scientists have criticized corporations for failing to act in the interests of the people, and their existence seems to circumvent the principles of democracy, which assumes equal power relations between individuals in a society. In an April 29, 1938 message to the Congress, Franklin D. Roosevelt warned that the growth of private power could lead to fascism: Dwight D. Eisenhower criticized the notion of the confluence of corporate power and d̈e factoQ̈uestion: Why do corporations lobby for laws that impoverish lower class citizens? [CS] While it is true that corporations may lobby for laws or regulations that are favorable to their own interests, it is not necessarily the case that these laws or regulations would have negative impacts on lower class citizens. [CE] [NSP] Anti-corporate activism control the markets, that income inequality is increasing, and that corporations have gained too much power. As part of the political left, activists against corporate power and influence support a decreased income gap and improved economical equity. The defenders of corporations such as Ron Arnold highlight that governments do legislate in ways that restrict the actions of corporations (see Sarbanes-Oxley Act) and that lawbreaking companies and executives are routinely caught and punished, usually in the form of monetary fines. In addition, from the perspective of business ethics it might be argued that chief executives are not inherently more evil than Question: Why do corporations lobby for laws that impoverish lower class citizens? [CS] For example, a corporation may lobby for laws that reduce regulation on its industry, which could potentially lead to lower costs and higher profits for the corporation, but could also have negative consequences for workers or consumers. [CE] It is important to recognize that the relationship between corporations, lobbying, and public policy is complex, and it is not always clear how specific laws or regulations will impact different groups of people.In general, it is important for citizens to stay informed about the activities of corporations and to advocate for policies that benefit the common good.",

B.3 Details of evaluation dataset

Construction Step Following the evaluation setup of Gao et al. (2023), we retrieve the top 100 paragraphs from the Wikipedia corpus provided by KILT for each instance in the dataset. We retrieve paragraphs by utilizing a well-performing retriever model, `contriever-msmarco` (Izacard et al., 2021). When constructing the top 100 paragraphs, we initially populate the corpus set with gold annotations from the KILT benchmark and subsequently supplement the remainder with paragraphs retrieved from `contriever-msmarco`, ensuring that all gold paragraphs are in the top 100 paragraphs.

Dataset Statistics and Characteristics In Table 4, we present the statistics and characteristics of the datasets in KILT (Petroni et al., 2021) benchmark employed for evaluation. Datasets lacking evidence annotation, such as WNED-CWEB, WNED-WIKI, and AIDA CoNLL-YAGO, are excluded from the KILT benchmark.

C Experimental Results

C.1 Weight parameter λ

Table 6 shows the performance by changing the weight parameter λ in Equation 5.

C.2 NTP + NSP benefits more from the flexibility of [NSP] placement

To assess how much the flexibility of determining where [NSP] is placed affects performance, we compare scenarios where [NSP] is always generated at the first token of response to a given input query (freeze placement of [NSP]), and where [NSP] is generated when the model predicts as necessary (flexible placement of [NSP]). Under these conditions, though the absolute score itself is higher for NTP + NSP, the degradation by freezing the placement is generally higher for models trained with NTP + NSP whereas those trained with NTP exhibit consistent improvements (Table 13). This suggests that the training approach of NTP + NSP benefits more from the flexibility of [NSP] placement, where the flexibility is one of the key benefits when operating `Gen` and $\texttt{Emb}_{seq}$ in the output space. Specifically, results show that the freedom to choose the placement of [NSP] enables NTP + NSP-trained models to leverage the advantages of communication between $\texttt{Emb}_{seq}$ and `Gen` more effectively, optimizing performance through appropriate integration of external information. All numbers in Table 11.

C.3 Does the benefit of semiparametric token-sequence co-supervision still hold when replacing $\texttt{Emb}_{seq}$ trained via NSP to more general retrieval model?

To see whether the benefits of multi-task training still persits when we replace $\texttt{Emb}_{seq}$ of NTP to other general retrieval models, including the one that is widely known in out-of-domain (contriever-msmarco), in Table 14, we compare NTP + NSP with variations of NTP. We could see the NTP + NSP in most cases shows the highest performance over the three variations of NTP. Llama without parameter sharing tends to show the strongest performance among the variations of NTP. Such results suggest that co-supervision on a language model stabilizes the nonparametric embedding space, which we conjecture is due to the benefit from robust parametric space.

C.4 Effect of Batch Size

Results in Figure 6 (Numbers in Table 10) show the average performance of each metric over 8 datasets in KILT with different batch sizes per GPU. Results show the importance of increasing the batch size in NTP + NSP; performance of all metrics tends

Name	Task	Instance Num	Input Format	Output Format	Reference
Natural Questions (NQ)	ODQA	2,837	Question	Extractive	Kwiatkowski et al. (2019)
Wizard of Wikipedia (WoW)	Dialogue Conversation	3,054	Long	Abstractive	Dinan et al. (2019)
FEVER	Fact Checking	10,444	Claim	Classification	Thorne et al. (2018)
TriviaQA	ODQA	5,359	Question	Extractive	Joshi et al. (2017)
ELI5	ODQA	1,507	Question	Long Abstractive	Fan et al. (2019)
Zero Shot RE (zsRE)	Slot Filling	3,724	Structured	Entity	Levy et al. (2017)
T-REx	Slot Filling	5,000	Structured	Entity	ElSahar et al. (2018)
HotpotQA	ODQA	5,600	Question	Short Abstractive	Yang et al. (2018)

Table 4: Statistic and characteristics of evaluation dataset.

Name	Metric
Natural Questions (NQ)	Answer Accuracy
Wizard of Wikipedia (WoW)	Unigram F1
FEVER	NLI
TriviaQA	Answer Accuracy
ELI5	RougeI
Zero Shot RE (zsRE)	Answer Accuracy
T-REx	Answer Accuracy
HotpotQA	Answer Accuracy

Table 5: Correctness metric for each datasets

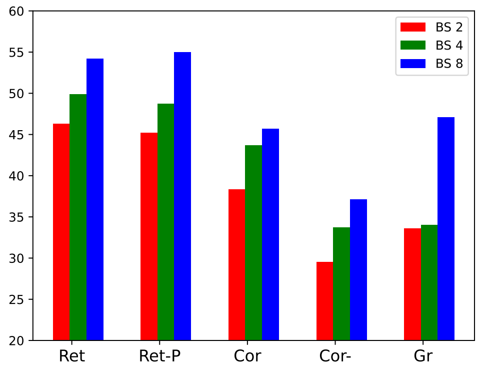

Figure 6: Average performance of each metric over 8 datasets in KILT when changing batch size. Each number indicates batch size per GPU.

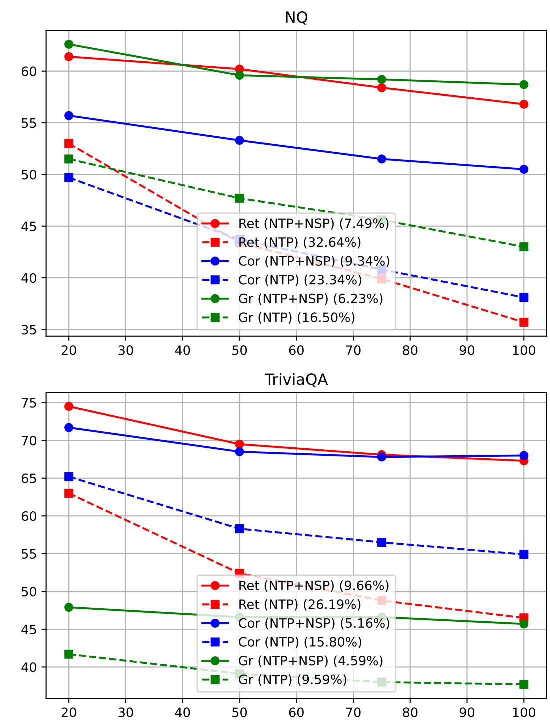

Figure 7: NTP + NSP tend to be more robust with larger corpus size (x-axis)

to increase with increasing batch size. We hypothesize such a trend largely due to stable training of NSP loss; as it is widely known that retrieval models tend to show higher performance with larger batch size, which is not always true for generation models (Keskar et al., 2017). As NSP loss is calculated via in-batch negatives, training with larger batch size in other words increasing the number of negatives consistently improves retrieval performance (Izacard et al., 2021; Karpukhin et al., 2020).

C.5 Performance gap tends to increase as corpus size increases

Figure 7 shows that the performance gap between NTP + NSP and NTP tends to increase as corpus size increases; NTP + NSP shows more stable and robust performance with different sizes of the corpus.

C.6 Grounding performance tends to vary by retrieval performance

When analyzing the impact of retrieval success on grounding performance, NTP + NSP significantly outperforms in grounding when retrieval is successful compared to when it fails, whereas the model trained only by NTP shows little difference regardless of retrieval outcome. Upon investigating why

Np weight	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R
	NQ*				WoW*				FEVER*				ELI5			
10^{-1}	59.0	55.9	47.7	43.3	42.0	41.6	15.2	36.8	67.5	64.7	64.4	36.0	**40.7**	29.0	22.5	5.2
10^{-2}	**65.1**	**62.7**	**55.7**	**62.6**	**49.8**	**49.7**	**15.7**	**63.7**	**77.5**	**75.9**	**65.2**	**28.0**	36.3	**29.4**	21.5	**8.7**
10^{-3}	33.7	33.6	35.7	34.9	14.0	14.0	13.2	27.0	27.4	27.3	28.3	9.2	19.9	19.4	**22.6**	4.2
	zsRE				T-REx				TriviaQA				HotpotQA			
10^{-1}	64.7	64.0	39.0	47.2	64.2	63.6	50.0	45.2	70.2	68.0	64.9	41.1	41.7	61.0	29.2	41.8
10^{-2}	**80.5**	**80.2**	**59.6**	**74.0**	**75.5**	**75.4**	**67.1**	**63.9**	**74.5**	**73.0**	**71.7**	**47.9**	**55.6**	**79.1**	**37.9**	**48.9**
10^{-3}	33.6	33.6	26.2	41.8	48.5	48.4	48.0	38.3	40.3	40.2	53.9	24.9	20.7	31.1	23.9	29.9

Table 6: Np weight (Top20)

		Cor.	Cor^-	Cor.	Cor^-	Cor.	Cor^-	Cor.	Cor^-
		NQ*		zsRE		T-REx		TriviaQA	
Top20	NTP	49.7	40.5	40.3	37.5	40.6	36.4	65.2	53.1
	NTP + NSP	55.7	49.3	59.6	58.1	67.1	62.5	71.7	65.2
Top100	NTP	38.1	27.7	27.1	24.3	30.8	25.8	54.9	39.2
	NTP + NSP	50.5	44.4	53.4	52.0	58.9	54.6	68.0	59.8

Table 7: Performance of correctness performance (Cor.) and correctness when considering those instances where retrieval fail as incorrect (Cor^-)

models trained with NTP + NSP exhibit lower grounding performance upon retrieval failure, it appears to stem from the disconnect caused by attempting to answer the query with the incorrect document. In such cases, the model might fetch information that seems closest to the expected answer from the external knowledge but ends up generating content that also contains knowledge from the given query, leading to less relevance to the retrieved paragraph or fabricating information not present in the paragraph (Examples in Table 12 in Appendix). Conversely, models trained with NTP demonstrate consistent grounding performance, unaffected by the success or failure of retrieval. This suggests that NTP's grounding capability is more reliant on its generative performance rather than the accuracy of retrieval, leading to similar outcomes irrespective of whether the correct information was retrieved or not. Based on these findings, future work could explore critiquing the success of retrieval based on grounding scores.

C.7 A single language model distinguishes and interprets both token and sequence embeddings space

The high performance of models trained through both supervision from parametric token embedding space and nonparametric sequence embedding space shows that a single model (`Gen`) can distinguish and comprehend both embedding spaces. As the task we experiment over requires the model to selectively draw distributions tailored to the appropriate embedding space as needed in the decoding step; when external knowledge is required, the output embedding operates with the nonparametric sequence embedding to find the most relevant one and for other cases, the output embedding operates with the parametric sequence embedding which generates in a natural form.

C.8 Generation performance without the condition of retrieval performance

As correctness and grounding performance depend on retrieval performance, we evaluate both NTP + NSP and NTP trained models in a setting where retrieval is always correct or always wrong to see the correctness and grounding performance without the condition of retrieval performance. We evaluate two settings where 1. oracle: retrieval is always correct and 2. failure: retrieval is always wrong. Table 15 shows the average performance of each metric over datasets in the KILT benchmark for oracle setting(all numbers in Table 8) and Table 9 shows performance for failure setting. **NTP + NSP trained models shows higher performance in the oracle setup whereas lower performance in the failure setup**. Such results correlate with the findings on top where since semiparametric token-sequence co-supervision trained tends to get the answer correct based on the retrieved paragraphs, it

		Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr
Force [NSP]			NQ*			WoW*			FEVER*			zsRE	
X	NTP	99.9	73.7	67.4	100.0	19.4	57.1	100.0	59.1	6.0	99.9	72.4	74.7
	NTP + NSP	99.8	68.6	69.3	100.0	19.6	69.8	100.0	65.4	31.0	100.0	71.7	80.8
O	NTP	100.0	71.9	69.5	100.0	19.3	59.4	100.0	59.1	6.0	100.0	70.3	74.8
	NTP + NSP	100.0	70.9	70.7	100.0	19.9	69.5	100.0	65.6	31.7	100.0	71.8	80.3
			T-REx			TriviaQA			ELI5			*Avg*	
X	NTP	98.0	58.0	56.0	96.7	73.2	48.9	100.0	20.7	6.5	99.2	53.8	45.2
	NTP + NSP	100.0	78.2	71.8	100.0	68.0	45.4	100.0	20.7	6.8	100.0	**56.0**	**53.5**
O	NTP	100.0	66.2	71.0	100.0	72.4	48.6	100.0	20.7	6.5	100.0	54.3	47.9
	NTP + NSP	100.0	78.9	72.0	100.0	68.4	44.2	100.0	20.7	6.3	100.0	**56.6**	**53.5**

Table 8: Performance over the oracle setup. We skip HotpotQA as the dataset requires two paragraphs, making it difficult to make in the same setting

		Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr
Force [NSP]			NQ*			zsRE			T-REx			TriviaQA			*Avg*	
X	NTP	0.0	12.2	13.8	0.0	7.9	15.7	0.0	9.3	7.9	0.0	7.6	8.3	0.0	**9.2**	**11.4**
	NTP + NSP	0.0	11.7	13.6	0.0	7.3	11.6	0.0	9.3	6.6	0.0	7.6	5.2	0.0	9.0	9.2
O	NTP	0.0	11.9	15.6	0.0	7.6	16.8	0.0	8.8	8.1	0.0	7.7	8.7	0.0	**9.0**	**12.3**
	NTP + NSP	0.0	11.9	13.6	0.0	7.3	11.7	0.0	8.9	7.1	0.0	7.4	5.0	0.0	8.9	9.3

Table 9: Experiment over the case where retrieval always fail. We skip HotpotQA as the dataset requires two paragraphs, making it difficult to make in the same setting, and datasets without answers as it is hard to distinguish false negatives.

BS	R	R-P	Cor	C^-	Gr	R	R-P	Cor	C^-	Gr	R	R-P	Cor	C^-	Gr	R	R-P	Cor	C^-	Gr
Top20																				
			NQ*					WoW*					FEVER*					ELI5		
2	63.1	59.1	50.0	44.6	60.6	37.5	36.8	14.4	7.3	33.5	71.4	67.5	56.3	40.7	6.0	38.9	27.0	21.8	8.4	8.7
4	63.3	59.0	50.8	44.9	49.0	40.8	40.1	15.3	8.2	31.7	73.7	69.2	66.0	50.0	21.0	35.1	25.0	22.2	7.8	5.3
8	65.1	62.7	55.7	49.3	62.6	49.8	49.7	15.7	10.4	63.7	77.5	75.9	65.2	51.5	28.0	36.3	29.4	21.5	7.8	8.7
			zsRE					T-REx					TriviaQA					HotpotQA		
2	68.8	65.9	50.8	49.4	48.1	68.0	66.5	60.0	55.0	46.5	71.4	68.9	57.8	54.6	43.0	48.9	71.0	29.2	26.4	43.1
4	78.0	75.2	56.4	55.1	50.5	73.0	71.3	65.0	60.1	46.5	72.7	68.9	68.5	62.7	38.6	52.2	73.6	36.8	33.9	36.8
8	80.5	80.2	59.6	58.1	74.0	75.5	75.4	67.1	62.5	63.9	74.5	73.0	71.7	65.2	47.9	55.6	79.1	37.9	35.2	48.9
Top100																				
			NQ*					WoW*					FEVER*					ELI5		
2	53.3	49.2	43.2	38.1	56.9	24.3	23.9	13.8	4.8	29.8	58.6	54.2	55.8	32.9	5.3	21.4	14.2	21.7	4.6	8.5
4	52.9	48.2	44.6	38.0	56.9	27.9	27.4	14.6	5.8	30.6	61.2	56.8	61.7	34.9	18.8	18.9	12.6	22.2	4.1	8.9
8	56.8	53.8	50.5	44.4	58.7	36.9	36.9	14.8	7.9	61.3	66.2	64.2	64.3	43.4	26.1	21.0	15.8	21.6	4.6	10.2
			zsRE					T-REx					TriviaQA					HotpotQA		
2	54.1	50.2	41.1	39.7	43.7	57.6	55.0	52.2	46.1	41.4	64.0	61.3	53.1	49.4	42.0	37.1	53.8	25.7	20.8	41.1
4	67.0	63.3	50.3	48.6	45.9	65.3	62.8	59.2	53.7	42.0	65.2	60.9	64.4	56.9	35.6	41.2	57.8	32.6	27.6	33.3
8	71.2	70.7	53.4	52.0	70.0	67.7	67.5	58.9	54.6	59.1	67.3	65.6	68.0	59.8	45.7	46.2	65.3	34.3	29.7	45.5

Table 10: Performance by changing batch size (BS).

shows high correctness in the oracle setup whereas lower correctness in the failure setup. Also as we could see that semiparametric token-sequence co-supervision trained models tend to show low grounding performance when retrieval fails, we can see that it shows lower grounding performance in the failure setup whereas high grounding performance in the oracle setup.

		Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr	Ret	Cor	Gr
	fix [NSP]	NQ*			zsRE			T-REx			TriviaQA			*Avg*		
NTP	X	35.7	38.1	43.0	32.8	27.1	47.2	47.4	30.8	45.1	46.5	54.9	37.7	32.5	30.2	34.6
	O	39.8	40.7	46.1	36.2	30.0	47.7	48.4	29.4	45.9	46.2	55.0	37.9	**34.1**	**31.0**	**35.5**
NTP + NSP	X	56.8	50.5	58.7	71.2	53.4	70.0	67.7	58.9	59.1	67.3	68.0	45.7	**52.6**	**46.2**	46.7
	O	56.2	50.2	59.4	70.8	52.7	70.0	67.8	60.2	60.0	66.7	67.0	45.6	52.3	46.0	**47.0**

Table 11: Performance when fixing [NSP] at the start token of response. We experiment over Top100.

Question	Retrieved Paragraph	Generated Response
who plays peter in what we do in the shadows	Steve Pemberton :: Steven James Pemberton (born 1 September 1967) is an English actor, comedian and writer, best known as a member of T̈he League of Gentlemenẅith Reece Shearsmith, Mark Gatiss, and Jeremy Dyson. Pemberton and Shearsmith also co-wrote and appeared in the sitcom P̈sychovilleänd the comedy-drama Ïnside No. 9.̈ His other television credits include D̈octor Who,̈ B̈enidorm,̈ B̈lackpool,̈ S̈hameless,̈ Ẅhitechapel,̈ Ḧappy Valleyänd M̈app and Lucia.̈	The actor Steve Pemberton plays the role of Peter in the British comedy television show.
the basic steps of rational decision making include defining goals and	Rational planning model :: The rational model of decision-making is a process for making sound decisions in policy making in the public sector. Rationality is defined as 2̌01ca style of behavior that is appropriate to the achievement of given goals, within the limits imposed by given conditions and constraints2̌01d. It is important to note the model makes a series of assumptions in order for it to work, such as:	The basic steps of rational decision making include defining goals and then analyzing alternative solutions to determine the best course of action.

Table 12: Example from NQ

	NTP			NTP + NSP		
	Ret	Cor	Gr	Ret	Cor	Gr
X	32.5	30.2	34.6	52.6	46.2	46.7
O	34.1	31.0	35.5	52.3	46.0	47.0

Table 13: Performance difference between static (fix the placement of [NSP]) and dynamic (allow the model to decide when to generate [NSP]) query

Retriever	Param Sharing	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R	Ret	Ret-P	Cor.	C-R
		NQ*				WoW*				FEVER*				ELI5			
Llama	x	53.3	50.3	**54.2**	28.0	28.3	28.1	**15.2**	48.0	61.9	60.7	58.2	5.7	18.5	14.9	**22.1**	7.3
Llama	o	35.7	34.4	38.1	43.0	14.7	14.6	13.4	40.0	31.4	30.4	56.7	5.2	12.7	11.0	21.9	7.0
contriever	o	50.3	49.8	51.0	52.1	23.0	23.0	14.9	42.5	39.0	38.9	54.5	11.0	18.8	9.6	**22.1**	7.3
NTP + NSP		**56.8**	**53.8**	50.5	**58.7**	**36.9**	**36.9**	14.8	**61.3**	**66.2**	**64.2**	**64.3**	**26.1**	**21.0**	**15.8**	21.6	**10.2**
		zsRE				T-REx				TriviaQA				HotpotQA			
Llama	x	49.8	49.5	38.4	53.9	52.4	52.4	33.2	45.7	65.1	64.1	66.4	44.8	43.1	63.3	34.0	39.3
Llama	o	32.8	32.7	27.1	47.2	47.4	47.4	30.8	45.1	46.5	45.8	54.9	37.7	12.6	12.4	19.6	11.8
contriever	o	45.0	44.0	36.2	42.2	36.9	36.7	33.8	33.1	**67.3**	65.1	**73.0**	42.2	46.0	65.1	34.0	37.4
NTP + NSP		**71.2**	**70.7**	**53.4**	**70.0**	**67.7**	**67.5**	**58.9**	**59.1**	**67.3**	**65.6**	68.0	**45.7**	**46.2**	**65.3**	**34.3**	**45.5**

Table 14: Top100

	Oracle		Failure	
	Cor	Gr	Cor	Gr
NTP	54.3	47.9	**9.0**	**12.3**
NTP + NSP	**56.6**	**53.5**	8.9	9.3

Table 15: Generation performance without the condition of retrieval performance