# Nearest Neighbor Zero-Shot Inference

**Weijia Shi** [1] **Julian Michael** [1] **Suchin Gururangan** [1] **Luke Zettlemoyer** [1]

## Abstract

We introduce kNN-Prompt, a simple and effective technique to use $k$-nearest neighbor ($k$NN) retrieval augmentation (Khandelwal et al., 2021) for zero-shot inference with language models (LMs). Key to our approach is the introduction of *fuzzy verbalizers* which leverage the sparse $k$NN distribution for downstream tasks by automatically associating each classification label with a set of natural language tokens. Across eleven diverse end-tasks (spanning text classification, fact retrieval and question answering), using kNN-Prompt with GPT-2 Large yields significant performance boosts over zero-shot baselines (14% absolute improvement over the base LM on average). Extensive experiments show that kNN-Prompt is effective for domain adaptation with no further training, and that the benefits of retrieval increase with the size of the model used for $k$NN retrieval. Overall, we show that augmenting a language model with retrieval can bring significant gains for zero-shot inference, with the possibility that larger retrieval models may yield even greater benefits.

## 1. Introduction

Retrieval-augmented language models (LMs) have access to a non-parametric memory, allowing them to directly access a large external text collection during inference. Previous work has shown that these models substantially outperform their non–retrieval-based counterparts on language modeling tasks (Khandelwal et al., 2020; He et al., 2021; Borgeaud et al., 2021), but it is an open question whether they also achieve similar gains in few-shot and zero-shot end task evaluations (Radford et al., 2019; Brown et al., 2020). In this paper, we demonstrate that, with some extensions to improve coverage, the performance gains of retrieval-augmented LMs generalize well to a wide range of downstream tasks.

We study the $k$-nearest neighbors language model (Khandelwal et al., 2020), which interpolates a neural LM's output distribution with a nearest-neighbor distribution constructed by retrieving tokens from a corpus using the LM's output

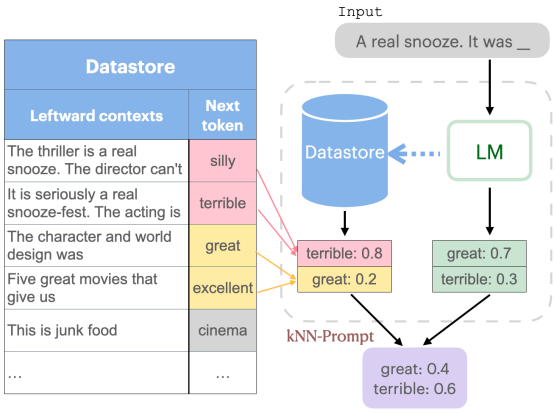


*Figure 1.* kNN-Prompt incorporates information from a large, heterogeneous corpus to facilitate zero-shot inference. The datastore contains key-value pairs where the key is an encoding of a leftward context and the value is the next token following the context.

embeddings. We are the first to study $k$NN-LM's zero-shot application to end tasks, and what we find is that applying the technique naïvely only produces marginal improvements. The main challenge is that the support of the $k$NN distribution is sparse (covering at most $k$ tokens, often less), as it only assigns probability mass to nearest neighbors. This means it often entirely misses the tokens that are used to verbalize the output label in the standard application of LMs to zero-shot classification: across the datasets we test, an output label receives nonzero probability under the $k$NN distribution only 45.8% of the time.

To address this challenge, we introduce kNN-Prompt, a simple and effective method built on $k$NN-LM for improving zero-shot inference with no further training. Key to our approach are *fuzzy verbalizers*, which automatically expand the set of tokens corresponding to each output label. For example, in Figure 1, the verbalized label of the negative sentiment is "terrible". Our fuzzy verbalizer also maps "silly" to negative sentiment, allowing the model to better leverage the information available in the $k$NN distribution.

Extensive experiments show that using kNN-Prompt with a heterogeneous datastore consistently improves an LM's zero-shot abilities on eleven tasks, including sentiment analysis, topic classification, entailment, fact retrieval and ques-

tion answering. These improvements hold for every model in the GPT-2 family. Furthermore, kNN-Prompt can be adapted to new domains and tasks with no further training. With a domain-specific datastore corpus, we achieve comparable or better performance to prompting the LM after domain-adaptive pretraining (**?**) on that corpus. To better understand these gains, we conduct a thorough analysis, showing that fuzzy verbalizers are essential for leveraging the $k$NN distribution, the benefits of retrieval increase with retrieval model size, and even relatively small datastores can yield sizeable performance gains if they are tailored to the domain or task.

## 2. Method

To perform zero-shot prediction on a downstream task using a pretrained language model, we recast the task as language modeling (Radford et al., 2019) by converting each input instance into a natural language prompt (subsection 2.1). We then augment the pretrained model with the $k$-nearest-neighbors language modeling technique from Khandelwal et al. (2020). To better benefit from the sparse $k$NN distribution, we introduce *fuzzy verbalizers* for mapping from the LM's outputs to a distribution over task-specific labels (subsection 2.3). Finally, we decode the output from this label distribution using the domain-conditional PMI scoring method of Holtzman et al. (2021).

### 2.1. Prompting and Verbalizers

We address classification problems where an instance consists of an input sequence of tokens $\mathbf{x} = (x_0, x_1, ..., x_{|x|})$ from a vocabulary $\mathcal{V}$ and an output label $y \in Y$. The output label set $Y$ may be fixed for the task (*text classification*) or provided for each instance as a set of expressions in $\mathcal{V}^*$ (*multiple-choice*). For example, in the sentiment analysis example in Figure 2, the input is $\mathbf{x}$ = "Mr. Tsai is one of world cinema's most gifted artists." The output labels are $Y$ = $\{y^+, y^-\}$, referring to positive and negative sentiment.

To cast the task as language modeling, we deterministically transform each input example $\mathbf{x}$ into a **prompt** $p(\mathbf{x})$. Providing this prompt to an LM yields a probability distribution

$$P_{LM}(\mathbf{z} \mid p(\mathbf{x})) = \prod_{z_i} P_{LM}(z_i \mid p(\mathbf{x}), \mathbf{z}_{<i})$$

over continuation sequences $\mathbf{z} \in \mathcal{V}^*$. To extract an output label from these continuations, we apply *verbalizers* $V : y \rightarrow \mathcal{V}^*$ (**?**) which map each output label $y \in Y$ to a natural language expression $V(y) = \mathbf{z}$. We can then compute a probability for each label:

$$P(y \mid \mathbf{x}) \propto P_{LM}(V(y) \mid p(\mathbf{x})), \tag{1}$$

normalizing over all $y \in Y$.

For example, our prompt transformation for sentiment analysis adds *It was* after the input, and uses the verbalizer $V(y^+) = \textit{great}, V(y^-) = \textit{terrible}$, which classifies sentiment according to the relative probabilities of *It was great* and *It was terrible* after the input sequence. In the case of multiple-choice problems, our verbalizer is just the identity function.

### 2.2. $k$-Nearest Neighbors Language Modeling

Following Khandelwal et al. (2020), we augment the LM with a *datastore* from which it can retrieve tokens that inform its predictions, improving performance without further training.

The datastore is a key-value store generated by running the LM over a corpus of text. Each value is a token $w \in \mathcal{V}$ from the corpus, and its key is the vector hidden representation at the output layer of the LM running forward on the left context $\mathbf{c} \in \mathcal{V}^*$ (call this $f(\mathbf{c})$). At inference time, when predicting the next token for an input sequence $\mathbf{c}$, the $k$NN-LM retrieves the $k$ nearest neighbors of $\mathbf{c}$ from the datastore according to the distance $d(\cdot, f(\mathbf{c}))$ of their key vectors.[1]

A softmax over the (negative) distances induces a distribution over the the tokens $w_i$ in the nearest neighbor set:

$$P_{k\text{NN}}(v \mid \mathbf{c}) \propto \sum_{(f(\mathbf{c}_i), w_i)} 1_{v=w_i} e^{\frac{-d(f(\mathbf{c}_i), f(\mathbf{c}))}{t}} \tag{2}$$

where $t$ is a temperature parameter.[2] We can then interpolate this with the original LM as follows:

$$P_{k\text{NN-LM}}(v \mid c) = (1-\lambda) P_{\text{LM}}(v|c) + \lambda P_{k\text{NN}}(v|c).$$

The hyperparameters for the $k$NN-LM approach are the number $k$ of nearest neighbors, the interpolation constant $\lambda$, the temperature $t$, and the choice of datastore.

### 2.3. Fuzzy verbalizers

One challenge in performing zero-shot inference with LMs on downstream tasks is the choice of verbalizer. On one hand, LMs may be highly sensitive to the particular surface form in ways that are irrelevant to the classification task (Holtzman et al., 2021). On the other hand, for a $k$NN model, the $k$ nearest neighbor set is sparse and may fail to cover any of the tokens in the set of verbalizers (i.e., $P_{kNN}(V(y)) = 0$ for all $y \in Y$), limiting its utility in those cases. To address these issues, we introduce *fuzzy verbalizers*, which associate each label $y$ with a neighborhood of token sequences around a specific verbalization $V(y) \in \mathcal{V}^*$.

[1] In this work, we use Euclidean distance; Khandelwal et al. use its square.

[2] We have added the temperature adjustment in the softmax on top of Khandelwal et al.'s $k$NN-LM formulation.

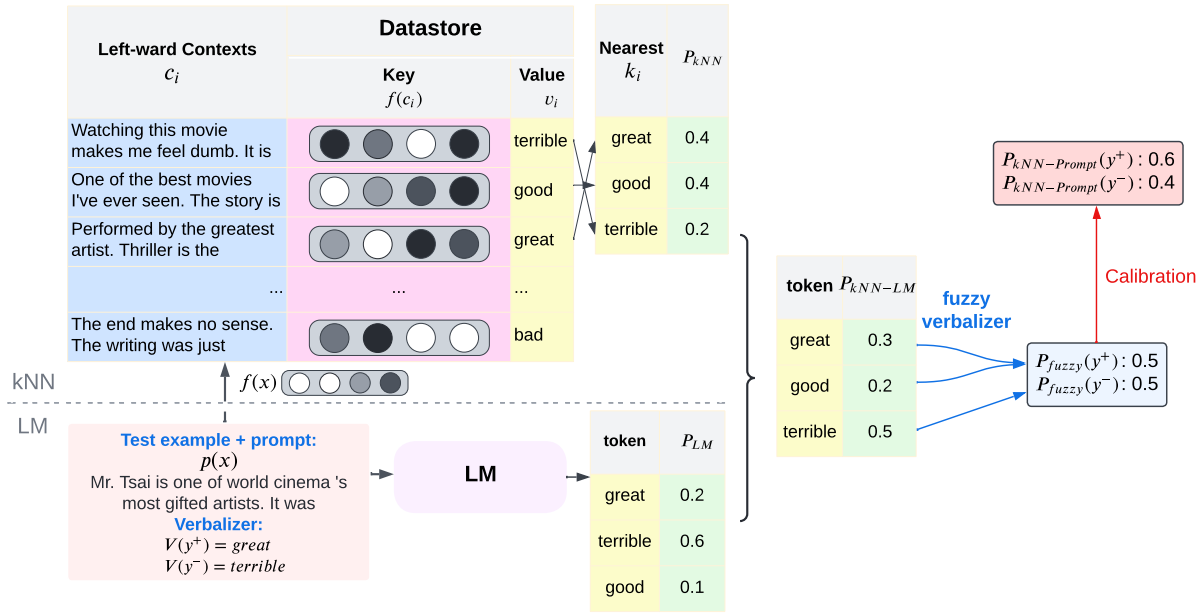


*Figure 2.* An illustration of kNN-Prompt applying to sentiment analysis tasks. Texts are encoded in the datastore, where each entry consists of a representation of a leftward context and its next token. During inference, a test example is mapped to a prompt form and used to retrieve the k most similar contexts and their next tokens from the datastore. kNN distribution is a multinomial computed on the distance of the text example and similar contexts. The final prediction is formed by combining the kNN distribution with the language model's output distribution.

To do this, we first associate each token $v \in \mathcal{V}$ with a neighborhood $\mathcal{N}(v) \subseteq \mathcal{V}$ of similar tokens. In particular, we use $v$'s top-5 most similar words according to the cosine similarity of their GloVe embeddings (**?**), as well as any of $v$'s synonyms in ConceptNet (Speer et al., 2017).[3] Then, for the purposes of calculating the probability of a verbalized label $\mathbf{z} = V(y)$, we treat a prediction of any token in each $z_i$'s neighborhood as a viable substitute for it, marginalizing over $\mathcal{N}(z_i)$ at each timestep:

$$P_{FV}(y \mid x) \propto \prod_{z_i \in V(y)} \sum_{v \in \mathcal{N}(z_i)} P(v \mid p(\mathbf{x}), \mathbf{z}_{<i}) \quad (3)$$

This incorporates more information from the LM to inform the induced distribution over labels $P_{FV}(y \mid x)$, and in the case of a $k$NN-based model, helps mitigate the effect the sparsity of the $k$NN distribution has on zero-shot prediction.

### 2.4. Full model

To make a zero-shot prediction for an input $\mathbf{x}$, we first transform it into a prompt $p(\mathbf{x})$ and obtain a distribution over continuations $(z)$ with a $k$NN-LM: $P_{kNN-LM}(\mathbf{z} \mid p(\mathbf{x}))$. We then convert this to a probability distribution over output labels $P(y \mid p(\mathbf{x}))$ using a fuzzy verbalizer (subsection 2.3, Equation 3). Finally, we output the best label according to

[3] https://conceptnet.io

| Corpus | Size | # Tokens |
|---|---|---|
| Wikitext-103 | 181MB | 114M |
| Amazon Reviews | 89MB | 19M |
| CC-NEWS | 457MB | 324M |
| IMDB | 45MB | 8M |
| Total | 722MB | 465M |

*Table 1.* Statistics of our heterogeneous datastore corpora.

the *domain-conditional PMI* scoring rule (Holtzman et al., 2021):

$$PMI_{DC}(y, p(\mathbf{x})) = \log \frac{P(y \mid p(\mathbf{x}))}{P(y \mid \mathbf{p})},$$

where $\mathbf{p}$ is a task-dependent string which is independent of the particular input (generally the local context at the end of the prompt, e.g., we use $\mathbf{p}$ = "It was" for sentiment analysis, as shown in Figure 2).

## 3. Experimental Setup

### 3.1. Tasks

We experiment with 11 tasks, including fact retrieval, question answering, topic classification, sentiment analysis, entailment and partisanship classification.

| | RTE | CQA | CB | Yahoo | LAMA | RT | SST-2 | CR | MR | HYP | AGN | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LM | 47.6 | 33.3 | 48.1 | 37.7 | 12.1 | 53.8 | 77.6 | 73.5 | 55.6 | 58.4 | 75.1 | 52.5 |
| LM+PMI | 54.6 | 46.5 | 52.0 | 45.6 | 18.3 | 78.9 | 74.1 | 67.3 | 78.7 | 51.2 | 66.9 | 57.6 |
| $k$NN-LM | 52.3 | 42.9 | 49.6 | 36.2 | 16.8 | 55.3 | 78.2 | 76.8 | 53.0 | 59.2 | 70.2 | 53.8 |
| kNN-Prompt | **54.9** | **49.2** | **57.2** | **53.4** | **29.5** | **84.1** | **87.8** | **84.7** | **82.3** | **60.5** | **77.1** | **66.6** |

*Table 2.* Zero-shot results on a variety of tasks. Our model, kNN-Prompt, handily outperforms Holtzman et al. (2021)'s PMI scoring method alone (LM+PMI) as well as the base $k$NN-LM method of Khandelwal et al. (2020).

### 3.2. kNN-Prompt Model Details

**Inference Model** For our main experiments, we directly use GPT-2 large from Huggingface[4] as our base LM.

**Retriever Model** Following the inference model, we use GPT-2 large to build the datastore. The keys are the 1280-dimensional hidden representations before the final MLP which predicts the token distribution at each timestep, produced using a single forward pass over the datastore corpus. For efficient similarity search, we create a FAISS (Johnson et al., 2019) index and search for nearest neighbors by Euclidean distance.

**Datastore Corpus** For our datastore, we aim to curate a large, heteregenous corpus of data broadly relevant to the tasks we evaluate. To this end, we combine four sources of data including Wikitext-103, the Amazon review corpus, and subsets of CC-NEWS[5] and IMDB[6] sampled uniformly from each. Table 1 lists the specifics of each data source.

**Inference Procedure** We retrieve k=512 neighbors, soften the kNN distribution with a temperature value of 3 and use an interpolation factor of $\lambda = 0.3$. Our primary evaluation is zero-shot. All hyperparameters were chosen on the basis of development experiments.

### 3.3. Baselines

**LM** is the result of prompting the base language model (GPT-2 Large), choosing the output label whose verbalizer has the highest probability under the language model $P_{LM}(V(y) \mid p(\mathbf{x}))$.

**LM+PMI** calibrates **LM** with domain-conditional PMI scoring (subsection 2.4).

**$k$NN-LM** directly applies the $k$NN-LM of Khandelwal et al. (2020) in the same way as **LM**, choosing the highest-probability output label.

[4]https://github.com/huggingface/transformers
[5]https://huggingface.co/datasets/cc_news
[6]https://datasets.imdbws.com

| | CR | HYP | LAMA |
|---|---|---|---|
| LM | $82.6_{4.1}$ | $59.0_{0.5}$ | $10.5_{2.4}$ |
| LM+PMI | $73.3_{5.5}$ | $58.8_{2.6}$ | $18.2_{3.7}$ |
| $k$NN-LM | $82.3_{4.2}$ | $58.9_{1.5}$ | $18.9_{1.6}$ |
| kNN-prompt | $\mathbf{84.8_{1.7}}$ | $\mathbf{63.4_{1.1}}$ | $\mathbf{29.2_{1.2}}$ |

*Table 3.* The mean and standard deviation for 4 uniformly sampled sets of 4 demonstration examples used for few-shot inference.

## 4. Experimental Results

Results for zero-shot prediction are in Table 2. kNN-Prompt outperforms all baselines in all tasks, improving over the base LM by 14.1% on average. The gains are particularly pronounced for MR and RT (sentiment analysis on movie reviews), Yahoo (topic classification), and LAMA (fact recovery). For MR and RT, the gains seem to come mostly from PMI calibration. On the other hand, large performance boosts on LAMA only come with the full kNN-Prompt model, which indicates the importance of combining retrieval, fuzzy verbalization, and PMI calibration for this task.

Interestingly, the $k$NN-LM alone yields a fairly small improvement over the base LM (about 1–2% on average). It is not strong enough to outperform LM+PMI even on LAMA, which intuitively should benefit from retrieval. This suggests that the fuzzy verbalizer and PMI calibration methods may help kNN-Prompt better leverage the information in the $k$-nearest neighbors distribution.

**Few-shot inference** For a subset of tasks, we additionally compare to a few-shot setting where we prepend four examples uniformly sampled from the training data to the input (Table 3). The demonstration examples are converted to prompt and verbalizer format. We report the mean accuracy and standard deviation with 4 different random seeds. We find that kNN-Prompt consistently outperform baselines, demonstrating that kNN-Prompt is applicable to the few-shot setting as well. We leave further exploration of this phenomenon to future work.

## 5. Conclusions and Future Work

We present kNN-Prompt, a new technique to augment LMs with nearest neighbor retrieval for zero-shot inference on end tasks. kNN-Prompt substantially improves zero-shot performance on a wide range of multiple-choice and classification tasks. With a domain- or task-relevant datastore, kNN-Prompt enables efficient domain adaptation with no additional training, and its benefits scale with the size of the retrieval model.

## References


Borgeaud, S., Mensch, A., Hoffmann, J., Cai, T., Rutherford, E., Millican, K., Driessche, G. v. d., Lespiau, J.-B., Damoc, B., Clark, A., et al. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2021.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

He, J., Neubig, G., and Berg-Kirkpatrick, T. Efficient nearest neighbor language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5703–5714, 2021.

Holtzman, A., West, P., Shwartz, V., Choi, Y., and Zettlemoyer, L. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7038–7051, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.564. URL https://aclanthology.org/2021.emnlp-main.564.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7 (3):535–547, 2019.

Khandelwal, U., Levy, O., Jurafsky, D., Zettlemoyer, L., and Lewis, M. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HklBjCEKvH.

Khandelwal, U., Fan, A., Jurafsky, D., Zettlemoyer, L., and Lewis, M. Nearest neighbor machine translation. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=7wCBOfJ8hJM.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Speer, R., Chin, J., and Havasi, C. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*, 2017.