Query-Focused Submodular Selection of Demonstrations for In-Context Learning

1st Paul Trust
School of Computer Science and Information Technology
University College Cork
Cork, Ireland
120222601@umail.ucc.ie

2nd Rosane Minghim
School of Computer Science and Information Technology
University College Cork
Cork, Ireland
r.minghim@cs.ucc.ie

***Abstract*—Adapting language models for downstream tasks has historically presented substantial challenges, demanding extensive datasets and computational resources for model training. However, the continuous evolution of these models, driven by pre-training on extensive internet data and the application of transfer learning techniques, has significantly mitigated the need for domain-specific training. We are now in an era marked by billion-parameter models, possessing substantial parametric knowledge, enabling their adaptation to diverse downstream tasks through human instructions or even a single example. This revolutionary approach commonly referred to as in-context learning (ICL), represents a recent paradigm shift within the field of natural language processing (NLP). ICL stands apart from conventional fine-tuning methods, as it empowers the adaptation of pre-trained models to new tasks without altering their core parameters or demanding updates. However, implementing ICL is not without its complexities; the mechanisms underlying in-context learning and the acquired knowledge remain elusive. Additionally, the criteria for selecting the most influential demonstration examples, pivotal for enhancing predictive performance, have yet to be comprehensively examined. Past research has favoured random selection to be select demonstration examples.**

Our contributions in this context is two-fold: First, we propose the use of query-focused submodular mutual information functions in the selection of demonstration examples for in-context learning. These functions enable the identification of demonstration examples that are both diverse and representative, ultimately bolstering few-shot performance compared to random and zero-shot baselines, as demonstrated in our experiments. Secondly, we offer a tool for interactive exploration, which aids in understanding the influence of hyperparameters. These include parameters like the number of demonstration examples and the methods used to generate them, especially regarding their effect on underlying data manifolds and clusters.

***Index Terms*—In-context Learning, Visualization, Language Models, Submodular Optimization, Data Selection**

I. Introduction

Machine learning models, particularly Large Language Models (LLM), have consistently demonstrated state-of-the-art performance across a wide range of tasks in numerous applications, including question answering, machine translation, text classification, and more [1], [2]. In the context of adapting machine learning models, the traditional pipeline typically involves the collection of data comprising training examples with input-output pairs for the target task. Subsequently, a model is trained on the dataset to make predictions on unseen examples. This well-established paradigm is commonly referred to as supervised learning and has played a pivotal role in machine learning tasks, with Natural Language Processing (NLP) being no exception [3].

However, it is worth noting that a fully supervised paradigm often proves insufficient for training high-quality models and there has been continuous research on efficient alternatives for training machine learning models. In the early stages of NLP, models heavily relied on feature engineering to encode domain knowledge and extract relevant features from raw data. This feature engineering process equipped models with the necessary inductive bias to effectively learn from limited data [4]. The advent of neural network models for NLP has led to joint learning of salient features alongside the training of the model itself [5], [6]. However, such fully supervised learning models are now playing a shrinking role, with the recent success of most models majorly explained by transfer learning. Models are first pre-trained as language models predicting the probability of observed textual data on very large raw text data to learn general-purpose features of the language. The pre-trained model can be adapted to different downstream tasks by introducing additional parameters and fine-tuning them using task-specific objective functions [7].

With the scaling of model size and corpus size, Large Language Models (LLMs) have demonstrated the ability to learn from just task instructions written in human language or from a few examples. Many studies have shown that LLMs can perform a series of complex tasks without the need for further re-training [8]–[10]. Using only test examples and human instructions, without the inclusion of demonstration examples, commonly referred to as "Prompting," has registered significant performance, especially for very large parameter language models. However, medium-sized languages still struggle to understand tasks solely based on task descriptions, and their performance is improved significantly by augmenting the instructions with a few demonstration examples, a process known as in-context learning [8]. In-context learning (ICL) induces the model to perform a downstream task by inputting task instructions and a few demonstration examples.

This work was done with funding from Science Foundation Ireland (SFI) Center for Research Training in Advanced Networks and Future Communication project code, R19784

First, ICL requires a few demonstration examples to form a demonstration context. These examples are usually written in natural language templates. Then ICL concatenates a query question and a piece of demonstration context together to form a prompt, which is then fed into the language model for prediction. Different from supervised learning, ICL requires no gradient-based training and therefore allows a single model to immediately perform a wide variety of tasks. Performing ICL, therefore, relies solely on the capabilities that a model learned during pre-training. This has led to great of success in diverse applications [8]. Despite the practical and powerful ability of ICL, the nature of demonstration examples has been shown to be a critical factor in its downstream [11]. However, currently demonstration examples are selected randomly, and domain experts who would like to experiment and use ICL on their own datasets have been left behind.

We present a solution based on query-focused submodular functions for selecting demonstration examples for in-context learning. In this approach, our target (query) corresponds to the test input, while the training set comprises the dataset from which we seek to extract the most representative and diverse examples. These selected examples are subsequently merged with the test input to create a context that is then provided to the Large Language Model (LLM) for prediction. Our experiments confirm the effectiveness of the proposed method, demonstrating its competitive performance when compared to existing methods that rely on random examples or prompting for generating examples used in constructing prompt templates. In order to leverage understanding of the process, we introduce an interactive tool designed to facilitate experimentation with various hyperparameters associated with our methodology. This tool allows users to explore options such as the inclusion of demonstration examples, selection methods, and inference models. Moreover, it offers visualizations of the outcomes post-training, incorporating multi-dimensional projections and evaluation metrics like accuracy, precision, F1-score, and recall. With this versatile tool, domain experts can conduct comprehensive experiments involving different models, prompt templates, and a spectrum of demonstration example selections, enabling them to identify the most effective approach for their specific tasks.

II. Background

This section provides concepts to our proposed method.

A. Submodular Functions

Submodular functions are an appealing class of functions for data subset selection in real-world applications due to the diminishing returns property and their ability to model properties of a good subset such as diversity, representation, and coverage [12]–[14]. Given a budget b (the number of elements we would like to select), consider a set $V = \{1, 2, 3, ..., n\}$ of elements, referred to as the ground set, and a utility function $f : 2^V \rightarrow \mathcal{R}$ as a measure of the quality of a subset $X \subseteq V$. The optimization problem can be formulated as in Equation 1:

$$\max f(X) \text{ subject to } c(X) \leq b \tag{1}$$

Here, the objective is to find a subset X that maximizes f while adhering to the constraint that the size of the set is less than or equal to the budget b. A discrete function $f : 2^V \rightarrow \mathcal{R}$, which returns a real value for any subset $S \subseteq V$, is submodular if for every $A \subseteq B \subseteq V$:

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B) \tag{2}$$

Given $g \in V$ and $g \notin B$, an alternative definition of submodularity is:

$$f(g \cup A) - f(A) \geq f(g \cup B) - f(B) \tag{3}$$

Equation 3 highlights the diminishing returns property of submodular functions, meaning that adding element g to set B yields a smaller gain in the target function than adding g to a smaller set A.

A greedy algorithm can be employed to optimize a submodular function when selecting a subset, offering a lower bound performance guarantee within a factor $(1 - 1/e)$ of the optimal solution. In practice, these greedy solutions are often within a factor of 0.98 of the optimal solution [15], [16]. This makes it preferable to frame the objective for data selection as a submodular function.

B. Submodular Mutual Information Functions

While submodular functions are a good choice for standard data selection, in this work we want to not only select the most informative and diverse set of points but also select points that are similar to a specific target slice (typically those that are close to a test input).

Formally, denote V as the ground set of items where we want to perform data selection. We denote Q an auxiliary set that contains user-provided information such as a query (for query-focused summarization or targeted subset selection). The auxiliary information provided by the user may not be in the same space as the items in V, for example, if the items in V are images, the query could be text queries. In such a scenario, we assume we have a joint embedding that can represent both the query and image items and correspondingly. We can define the similarity between the items in V and Q. Next, let $\Omega = V \cup Q$ and define a set function $f : 2^\Omega \rightarrow \mathcal{R}$. Although f is defined on Ω, selection on the items in V, the discrete optimization problem will be only on subsets of V.

The submodular mutual information (SMI) functions to capture the second property of submodular functions

$$I_f(S, Q) = f(S) + f(Q) - f(S \cup Q) \tag{4}$$

where $S, Q \subseteq V$, Q is the query or target set, in our case the test input to be fed into the generative model for inference. Intuitively, maximizing the SMI functions ensures that we obtain diverse subsets that are relevant to a query set Q. It is also easy to see that $I_f(S, Q)$ is equal to the mutual

information between two random variables when f is the entropy function. Thus, this measures the similarity between Q and S where Q is the query set.

Examples of submodular Mutual information Functions include the following: For any two data points $i \in V$ and $j \in Q$, let s_{ij} denote the similarity between them.

Graph Cut MI. The submodular mutual information (SMI) instantiation of graph-cut (GCMI) is defined as:

$$T_f(S;Q) = 2\lambda \sum_{i \in S} \sum_{j \in Q} s_{ij} \tag{5}$$

Since maximizing GCMI maximizes the joint pairwise sum with the query set, it will lead to a subset similar to the query set Q. GCMI models only query-relevance and does not select based on diversity [17].

Facility Location MI: The facility location mutual information (FLMI) function [17] takes the expression:

$$I_f(S;Q) = \sum_{i \in Q} \max_{j \in S} s_{ij} + \eta \sum_{i \in S} \max_{j \in Q} s_{ij} \tag{6}$$

FLMI is very intuitive for query relevance as well. It measures the similarity between the representation of data points that are the most relevant to the query set and vice versa.

Log Determinant MI: The SMI instantiation of Log Determinant MI can be defined as follows. Given a submodular mutual information function, Log Determinant Mutual information is its instantiation using the Log Determinant function.

Let $S_{A,B}$ be the cross-similarity matrix between the items in sets A and B. Also, denote $S_{AB} = S_{A \cup B}$.

We construct a similarity matrix S^{η} (based on matrix S) in such a way that the cross-similarity between A and Q is multiplied by η (i.e., $S^{\eta}_{A,Q} = \eta S_{A,Q}$) to control the trade-off between query relevance and diversity. Higher values of η ensure greater query-relevance, while lower values favor diversity.

Using the similarity matrix defined above and with $f(A) = \log\det(S^{\eta}_A)$, we have:

$$I_f(A;Q) = \log\det(S_A) - \log\det\left(S_A - \eta^2 S_{A,Q} S_Q^{-1} S_{A,Q}^T\right) \tag{7}$$

III. Related Work

A. In-context Learning

With the increasing ability of large language models (LLMs) [8], [18], [19], in-context learning (ICL) has become a new paradigm for natural language processing, where LLMs make predictions only based on contexts augmented with a few examples. Many studies have shown that LLMs can perform a series of complex tasks through ICL, such as solving mathematical reasoning problems [20]. The key idea of in-context learning is to learn from analogy. First, ICL requires a few examples to form a demonstration context. These examples are usually written in natural language templates. ICL concatenates a query and a piece of demonstration context together to form a prompt, which is then fed into the language model for prediction. Different from supervised learning that requires a training stage that uses backward gradients to update model parameters, ICL perfoms predictions directly on the pre-trained model without conducting any parameter updates.

This paradigm of ICL has multiple attractive advantages, the demonstration is written in natural language which provides an interpretable interface to communicate with language models [10], and thus human can easily inject domain knowledge by changing the demonstrations and templates. Most importantly, ICL is a training-free learning framework which makes it appealing since it has the potential to greatly reduce the computation costs for adaptation to new tasks [21].

Despite all the impressive and promising performance of ICL, some problems still exist, ICL is still out-performed by full fine-tuning approaches and several studies have observed that the ICL ability is improved significantly via adaptation during pre-training [22], [23]. In addition, the performance of ICL is sensitive to a specific setting, including the selection of in-context examples [24] and ordering of examples among others [25]. Our work investigates the use of successful targeted data selection approaches based on query-focused submodular information functions to select both diverse and relevant demonstration examples for the construction of the prompt template. Moreover, we introduce an interactive tool to enable selecting different hyperparameters during in-context learning.

B. Visualization for NLP

Visualization methods and tools for interacting with language models have gained increasing popularity alongside the rise of the models themselves. These tools serve several functions, including the explainability of models and understanding their internal workings. In the realm of character-level recurrent neural networks, early work [26] demonstrated the effectiveness of hidden states in capturing unique patterns in the training text. Compositionality in neural linguistic models has been examined by utilizing techniques inspired by model interpretability techniques from computer vision [27]. Various tools have been developed to visualize the hidden states and training process of recurrent neural networks, such as RNNVis [28], RNNbow [29], and LSTMViz [30]. Word embedding and sentence embeddings, like Word2Vec [31] and Sentence Transformers [32], illustrate how semantic relationships can be recovered by projecting into high-dimensional embedding spaces.

Several tools have also been proposed to visualize and understand transformers. ExBERT [33] is a flexible tool designed to help humans conduct interactive investigations and formulate hypotheses for model-internal reasoning processes. Other tools explore and compare the behaviour of model-internal distributions during text generation, such as LMDIFF [34] and GLTR [35].

The most similar to our work is that by Strobelt et al. (2022) [36] who introduced PromptIDE, a tool that allows users to experiment with various prompt variations, visualize prompt performance, and iteratively optimize prompts. These visualization tools and studies not only demonstrate the benefits of

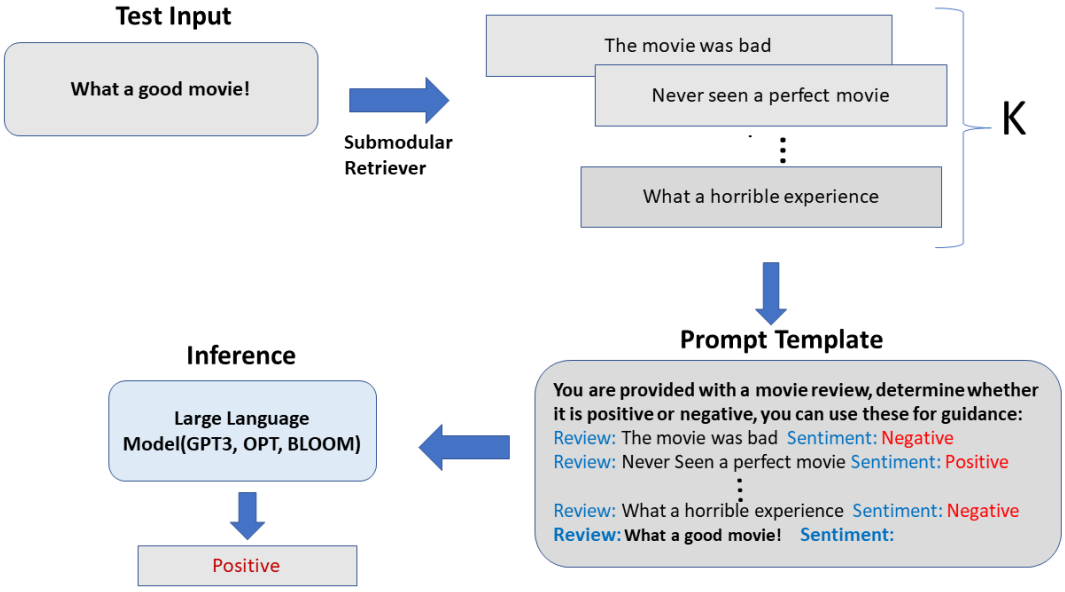


Fig. 1: When presented with a test input for which we seek a prediction, typically a label in our case, we employ a query-focused submodular function to select relevant examples from our training dataset. This submodular function is designed to choose examples that are both diverse and representative. The resulting set of k selected examples is then concatenated with the test input, serving as context. This augmented context is subsequently sent to the model for the purpose of inference, allowing us to obtain the desired prediction.

visualization for understanding NLP models but also reveal the enormous potential for future applications. Our tool is different from PromptIDE in that we do not only provide the user with the flexibility to write prompts but also facilitate in-context learning through demonstration selection.

IV. Methods

A. Description of Methodology

Consider a training set of N labeled examples, denoted as $D = \{(x_i, y_i)\}^N$, consisting of input-output sequences. Here, x_i represents the text input, and y_i is the corresponding label. Additionally, let's assume we possess a test dataset containing M examples without labels, on which we wish to perform inference. We denote this test dataset as $D_{\text{test}} = \{(x_{\text{test}})\}^M$. Our methodology commences with the training of a retriever model, denoted as $R(x_{\text{test}}, D)$. This retriever model employs query-focused submodular functions to sample a subset of k training examples, represented as $\{(x_i, y_i)\}_{i=1}^k \subset D$. Subsequently, these k retrieved examples are concatenated with the context for all M examples in the test dataset to be used for inference. Our methodology consists of two distinct steps: the retrieval step and the inference step, each of which is elucidated in greater detail in the following sections.

1) Retrieval Stage: In this step, we aim to select representative and diverse in-context demonstration examples from our training data. Both examples in the test and training datasets are embedded using the state-of-the-art sentence transformer [37]. From the N training examples, we select k examples by leveraging Submodular Mutual Information (SMI) functions for this specialized data selection. Let the embedding of the test input we're trying to label be represented by T, and U denote the chosen N examples. Utilizing a suitable feature representation for these instances (in our case sentence embeddings), we evaluate kernels of similarities within U, T, and between both. This helps in forming a MI function $I_f(A;T)$. We aim to identify the best subset $\tilde{A} \subseteq U$ of size k, with T serving as our reference set, by optimizing this function. For this purpose, we employ diverse MI functions, including but not limited to Facility MI, Graph Cut MI, and Log Determinant MI.

The finalized k examples are integrated into a prompt template, C. This can either be a combination of an optional task directive I and the k demonstrations, presented as $C = \{I, s(x_1, y_1), \ldots, s(x_k, y_k)\}$, or simply the demonstrations on their own: $C = \{s(x_1, y_1), \ldots, s(x_k, y_k)\}$. Here, each instance $s(x_k, y_k, I)$ is articulated in natural language, aligning with the task's requirements.

2) Inference Stage: With the context given by $C = \{I, s(x_1, y_1)\}$ which includes k demonstration examples and the test input, we feed this data into a pre-trained language model to deduce the corresponding label. For determining the labels, we employ the metric of perplexity [2]. Specifically, Perplexity (PPL) evaluates the sentence perplexity of the complete input sentence $S_j = \{C, s(x, y_j, I)\}$, made up of demonstration tokens C, the input query x, and a prospective label y_j. The pre-trained model M predicts the label based

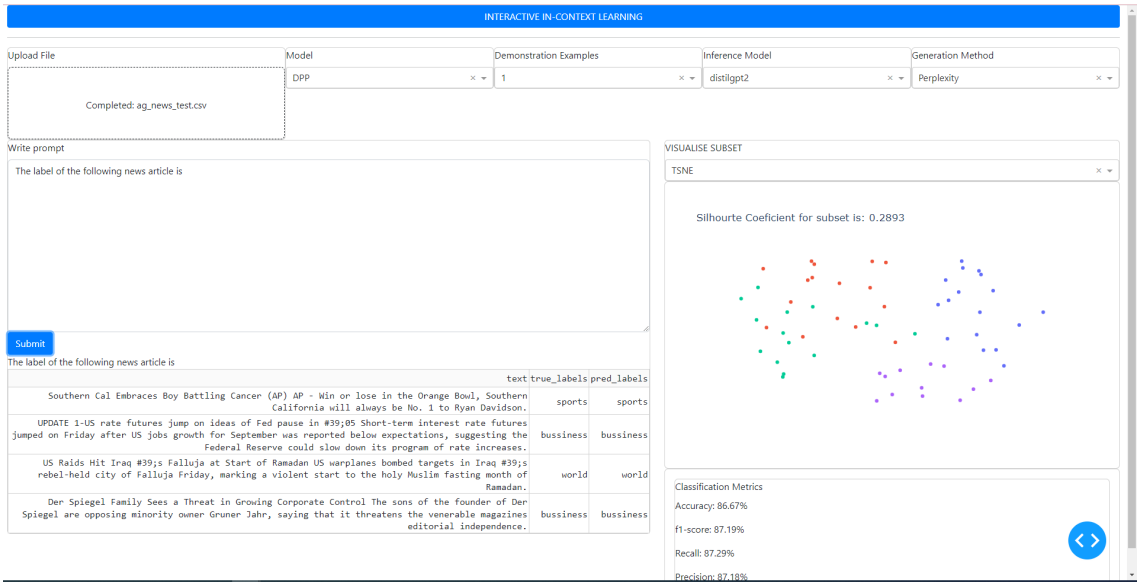


Fig. 2: To start with the tool, users should first upload their desired dataset. This dataset should contain a column for text intended for classification and another for the corresponding labels. After uploading, users can craft a suitable prompt to pair with demonstration examples. Subsequently, users can choose from a range of retrieval models, including options like random, prompt, DPP, Facility, and Graph Cut. Selections can also be made regarding the quantity of demonstration examples, the inference model, and the generation method. After finalizing their choices and letting the tool produce predicted labels, users will see a preview of five entries showcasing both actual and predicted labels. For a comprehensive understanding of label relationships, users can utilize multidimensional projections to visualize the predicted labels. Additionally, the tool offers classification metrics, giving insights into the model's performance metrics, including accuracy, precision, recall, and F1 score.

on the highest-scoring candidate, taking the demonstration C into account. The probability of a particular answer y_j can be framed through a scoring function f using the model M:

$$p(y_j|x) = f_M(y_j, C, x) \tag{8}$$

The ultimate label, $\tilde{y}$, is chosen as the candidate with the peak likelihood:

$$\tilde{y} = \underset{y_j \in Y}{\operatorname{argmax}}\, p(y_j|x) \tag{9}$$

Function f provides an estimation of a candidate answer's likelihood, based on both the demonstration and the input query.

B. Interactive System for In-context Learning

In addition to proposing submodular mutual information functions for selecting both diverse and relevant examples to be used as demonstration examples in prompt templates, we also propose an interactive system that enables domain experts to perform in-context learning.

The **dataset and prompt section**, as shown in Figure 2, enables users to upload datasets they want to perform inference on. If they upload only one dataset, then we divide it into train, test, and validation splits. The **write prompt** section enables the user to describe the task in a human-understandable way; for example, for text classification, the template could be written as "What is the topic of the given text?: {label}". Using the prompt written by the user, we concatenate the number of demonstration examples to construct a prompt template to be used for generation.

Users have the flexibility to choose between retrieval models, with choices being random, prompt, and several submodular mutual information functions like facility, graph cut, and DPP. They can also select the number of demonstrations to be used and the inference model for generation.

The **sample dataset** view enables the user to see a sample of documents alongside their actual labels and predicted labels. We also have a **results section** where the user can visualize the predicted labels alongside the documents with multidimensional projections. The user can choose to visualize the documents using t-SNE, UMAP, or PCA. The last part of the results section enables the user to view aggregated results on the test set in terms of accuracy, F1-score, precision, and recall.

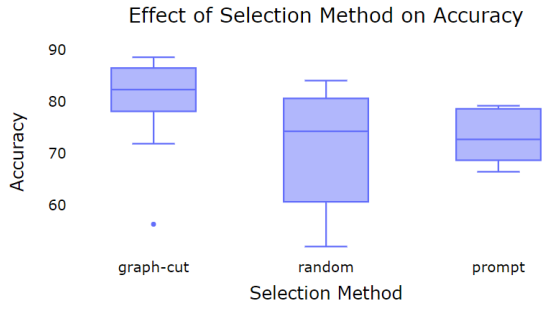
V. Experimental Setup

A. Data

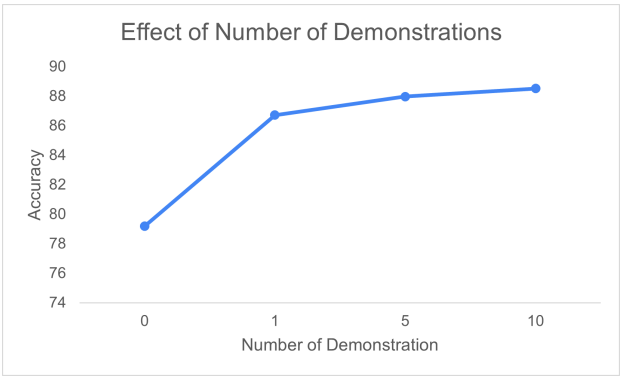
We apply our proposed method to the following tasks; two sentiment classification datasets: SST-2 and SST-5 [38], topic classification datasets: AgNews [39]. We report accuracy, f-score, precision and recall on the datasets.

B. Baseline Methods

We compare our framework with three baselines:**Random Sampling**: For each test sentence, we randomly select in-context example from the training set. We refer to this method



(a) Retrieval Method



(b) Number of demonstrations

Fig. 3: A box plot illustrating the impact of various retrieval methods on accuracy, alongside a line graph depicting the influence of the number of demonstrations on accuracy, effectively communicates the findings. The reported results, based on an average accuracy from 50 experiments, clearly demonstrate the effectiveness of our proposed methodology in enhancing text classification performance.

Retrieval Method	SST-2	SST-5	AgNews
Prompting	51.63	25.88	26.13
Random-1shot	55.56	19.82	31.38
Facility MI-1shot	64.90	34.61	88.80
DPP MI-1shot	64.80	35.62	89.62
Graphcut MI-1shot	64.78	34.61	89.60
Random-5shot	51.95	24.12	27.47
Facility MI-5shot	83.64	37.15	87.99
DPP MI-5shot	**84.35**	36.06	89.13
Graphcut-5shot MI	84.24	**37.51**	**90.56**

TABLE I: Evaluation results for different retrieval methods using the same inference model. Numbers in bold indicate the highest accuracy among all methods

as Random in the experimental results. **Prompting**: Prompting [2] is a special case of ICL without in-context examples. This can be seen as zero-shot learning or in-context learning with zero examples.

C. *Inference Models*

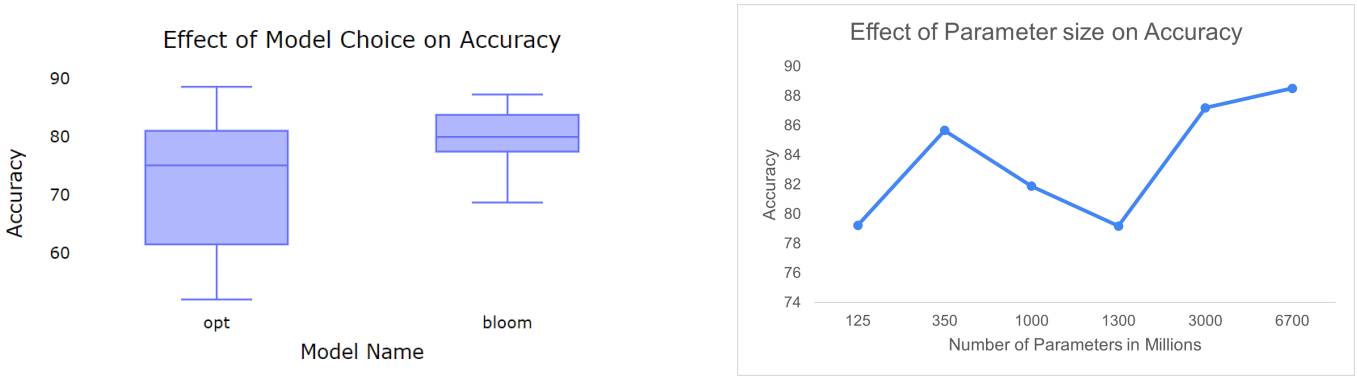
We employed three open-source pre-trained models for our research: GPT-2 [40], OPT [19], and BLOOM [9]. These models have performance on par with GPT-3 and are freely accessible. OPT [19], with parameters from 125 million to 175 billion, was trained on diverse datasets like the Pile [41] and BooksCorpus [42]. BLOOM [9], with 176 billion parameters, is based on the ROOTS corpus [43]. GPT-2 [40], spanning from 117 million to 1.5 billion parameters, has shown excellence in various NLP tasks.

VI. Results

In Table I, we find that in-context learning methods generally outperform prompting. However, the selection of in-context examples plays a vital role in performance. For instance, poor in-context examples, such as random baselines, can lead to worse performance than the prompting baseline on some datasets like SST-5. In this case, the random baseline in a 1-shot setting has an accuracy of 19.82%, which is lower than the 25.88% achieved by prompting alone. However, this trend is not universal across all datasets, as random baselines perform better than prompting on datasets like SST-2 and AgNews in a 1-shot setting. Increasing the number of in-context examples typically improves accuracy for most tasks, but this is not a guarantee for all methods. For the random baseline on the SST-2 dataset, performance decreases when the number of in-context examples is raised from 1 to 5, resulting in a 3.61% drop in accuracy. A similar decline is observed with the AgNews dataset.

On the other hand, increasing the number of in-context examples for other datasets and employing our query-based submodular mutual information functions leads to performance gains. For example, on the SST-5 dataset, even with the random baseline, we observe a 4.3% increase. The same trend is observed with AgNews and SST-5 when using Facility MI, DPP MI, and Graph Cut MI functions. These findings highlight the importance of intelligent demonstration selection on the performance of a model. Randomly selected examples may hinder performance as the number of demonstration examples increases, unlike when demonstration examples are carefully chosen. We also note that performance plateaus after reaching five demonstration examples for some datasets like AgNews and SST-5. Further increasing the number of demonstration examples may negatively impact the generative model's performance.

While intelligent selection demonstrates competitiveness compared to random selection in Table I, we investigate whether a significant difference exists among query-focused submodular information functions. Our analysis of the AgNews, SST-5, and SST-2 datasets reveals that no single function consistently outperforms the others across all tasks, and the performance difference between these functions is minimal across various datasets. We further conducted over 50 experiments on SST-2 datasets using different models, including OPT, BLOOM, with various model sizes. Figure



(a) Inference Mode

(b) Number of Parameters

Fig. 4: A box plot displaying the influence of different inference model choices on accuracy, complemented by a line graph demonstrating the impact of the number of parameters on accuracy. Model sizes are denoted in millions. These visualizations effectively convey the findings, with the reported results showcasing an average accuracy derived from 50 experiments

3a demonstrates that the choice of retrieval method is crucial. Graph cut, a submodular mutual information function, obtains a maximum accuracy value of approximately 88%, while random achieves a maximum value of around 78%, and random selection obtains a maximum value of around 84%. Figure 3b further shows that the best performance reported increases with the number of demonstrations, but our experiments revealed that beyond 10 demonstration examples, the difference in accuracy was small.

Figure 4a illustrates that different models, such as GPT-2, BLOOM, and OPT, exhibit varying performances, with OPT achieving the maximum performance in the 50 experiments. However, the differences could partly be attributed to the varying model sizes since not all models were equally sized, and there are no equally open-source pre-trained models for all three models. In our experiments, OPT (the largest model used, with 6.7 billion parameters) and BLOOM (the largest model used, with 3 billion parameters) were larger than GPT-2. Figure 4b indicates that although increasing the number of parameters led to the best performance, this was not a consistent behavior, as billion-parameter models were outperformed by models with 350 million parameters.

VII. Conclusion

We propose submodular mutual information functions to choose diverse and relevant demonstration examples for in-context learning. Our experiments indicate that our approach is superior to random sampling and traditional prompting methods emphasizing our initial hypothesis that the success of in-context learning is largely influenced by the quality of demonstration examples. While adding more demonstrations generally enhances performance, there are diminishing returns, particularly with random picks. We also note that the effectiveness of submodular information functions is task-dependent, suggesting there's no universal function suitable for all datasets. Contrary to initial assumptions, the size and type of the model don't always guarantee better performance for example smaller models sometimes obtained better performance compared to the large alternatives in the same family of models. Additionally, we developed a tool to experiment with various hyper-parameters in in-context learning, such as the number of demonstration examples, selection techniques, and inference models.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

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

[3] S. B. Kotsiantis, I. Zaharakis, P. Pintelas *et al.*, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007.

[4] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine learning*, vol. 46, pp. 389–422, 2002.

[5] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *Advances in neural information processing systems*, vol. 13, 2000.

[6] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of machine learning research*, vol. 12, no. ARTICLE, pp. 2493–2537, 2011.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://aclanthology.org/N19-1423

[8] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[9] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé *et al.*, "Bloom: A 176b-parameter open-access multilingual language model," *arXiv preprint arXiv:2211.05100*, 2022.

[10] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Comput. Surv.*, vol. 55, no. 9, jan 2023. [Online]. Available: https://doi.org/10.1145/3560815

[11] A. Webson and E. Pavlick, "Do prompt-based models really understand the meaning of their prompts?" in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 2300–2344. [Online]. Available: https://aclanthology.org/2022.naacl-main.167

[12] E. Tohidi, R. Amiri, M. Coutino, D. Gesbert, G. Leus, and A. Karbasi, "Submodularity in action: From machine learning to signal processing applications," *IEEE Signal Processing Magazine*, vol. 37, no. 5, pp. 120–133, 2020.

[13] F. Bach *et al.*, "Learning with submodular functions: A convex optimization perspective," *Foundations and Trends® in Machine Learning*, vol. 6, no. 2-3, pp. 145–373, 2013.

[14] F. Bach, "Submodular functions: from discrete to continuous domains," *Mathematical Programming*, vol. 175, pp. 419–459, 2019.

[15] S. Tschiatschek, R. K. Iyer, H. Wei, and J. A. Bilmes, "Learning mixtures of submodular functions for image collection summarization," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014. [Online]. Available: https://proceedings.neurips.cc/paper/2014/file/a8e864d04c95572d1aece099af852d0a-Paper.pdf

[16] V. Kaushal, R. Iyer, K. Doctor, A. Sahoo, P. Dubal, S. Kothawade, R. Mahadev, K. Dargan, and G. Ramakrishnan, "Demystifying multi-faceted video summarization: tradeoff between diversity, representation, coverage and importance," in *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2019, pp. 452–461.

[17] S. Kothawade, N. Beck, K. Killamsetty, and R. Iyer, "Similar: Submodular information measures based active learning in realistic scenarios," *Advances in Neural Information Processing Systems*, vol. 34, pp. 18 685–18 697, 2021.

[18] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. v. d. Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, "Training compute-optimal large language models," 2022. [Online]. Available: https://arxiv.org/abs/2203.15556

[19] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin *et al.*, "Opt: Open pre-trained transformer language models," *arXiv preprint arXiv:2205.01068*, 2022.

[20] S. Wang, Y. Liu, Y. Xu, C. Zhu, and M. Zeng, "Want to reduce labeling cost? GPT-3 can help," in *Findings of the Association for Computational Linguistics: EMNLP 2021*. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 4195–4205. [Online]. Available: https://aclanthology.org/2021.findings-emnlp.354

[21] T. Sun, Y. Shao, H. Qian, X. Huang, and X. Qiu, "Black-box tuning for language-model-as-a-service," in *International Conference on Machine Learning*. PMLR, 2022, pp. 20 841–20 855.

[22] S. Min, M. Lewis, L. Zettlemoyer, and H. Hajishirzi, "MetaICL: Learning to learn in context," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 2791–2809. [Online]. Available: https://aclanthology.org/2022.naacl-main.201

[23] Y. Chen, C. Zhao, Z. Yu, K. McKeown, and H. He, "On the relation between sensitivity and accuracy in in-context learning," *arXiv preprint arXiv:2209.07661*, 2022.

[24] J. Liu, D. Shen, Y. Zhang, B. Dolan, L. Carin, and W. Chen, "What makes good in-context examples for GPT-3?" in *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*. Dublin, Ireland and Online: Association for Computational Linguistics, May 2022, pp. 100–114. [Online]. Available: https://aclanthology.org/2022.deelio-1.10

[25] Z. Zhao, E. Wallace, S. Feng, D. Klein, and S. Singh, "Calibrate before use: Improving few-shot performance of language models," in *International Conference on Machine Learning*. PMLR, 2021, pp. 12 697–12 706.

[26] A. Karpathy, J. Johnson, and L. Fei-Fei, "Visualizing and understanding recurrent networks," *arXiv preprint arXiv:1506.02078*, 2015.

[27] J. Li, X. Chen, E. Hovy, and D. Jurafsky, "Visualizing and understanding neural models in nlp," *arXiv preprint arXiv:1506.01066*, 2015.

[28] Y. Ming, S. Cao, R. Zhang, Z. Li, Y. Chen, Y. Song, and H. Qu, "Understanding hidden memories of recurrent neural networks," in *2017 IEEE conference on visual analytics science and technology (VAST)*. IEEE, 2017, pp. 13–24.

[29] D. Cashman, G. Patterson, A. Mosca, N. Watts, S. Robinson, and R. Chang, "Rnnbow: Visualizing learning via backpropagation gradients in rnns," *IEEE Computer Graphics and Applications*, vol. 38, no. 6, pp. 39–50, 2018.

[30] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush, "Lstmvis: A tool for visual analysis of hidden state dynamics in recurrent neural networks," *IEEE transactions on visualization and computer graphics*, vol. 24, no. 1, pp. 667–676, 2017.

[31] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[32] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[33] B. Hoover, H. Strobelt, and S. Gehrmann, "exBERT: A Visual Analysis Tool to Explore Learned Representations in Transformer Models," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Online: Association for Computational Linguistics, Jul. 2020, pp. 187–196. [Online]. Available: https://aclanthology.org/2020.acl-demos.22

[34] H. Strobelt, B. Hoover, A. Satyanarayan, and S. Gehrmann, "Lmdiff: A visual diff tool to compare language models," *arXiv preprint arXiv:2111.01582*, 2021.

[35] S. Gehrmann, H. Strobelt, and A. M. Rush, "Gltr: Statistical detection and visualization of generated text," *arXiv preprint arXiv:1906.04043*, 2019.

[36] H. Strobelt, A. Webson, V. Sanh, B. Hoover, J. Beyer, H. Pfister, and A. M. Rush, "Interactive and visual prompt engineering for ad-hoc task adaptation with large language models," *IEEE transactions on visualization and computer graphics*, vol. 29, no. 1, pp. 1146–1156, 2022.

[37] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. [Online]. Available: https://aclanthology.org/D19-1410

[38] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, 2013, pp. 1631–1642.

[39] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *Advances in neural information processing systems*, vol. 28, 2015.

[40] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[41] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima *et al.*, "The pile: An 800gb dataset of diverse text for language modeling," *arXiv preprint arXiv:2101.00027*, 2020.

[42] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler, "Aligning books and movies: Towards story-like visual explanations by watching movies and reading books," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 19–27.

[43] H. Laurençon, L. Saulnier, T. Wang, C. Akiki, A. V. del Moral, T. L. Scao, L. Von Werra, C. Mou, E. G. Ponferrada, H. Nguyen *et al.*, "The bigscience roots corpus: A 1.6 tb composite multilingual dataset," *arXiv preprint arXiv:2303.03915*, 2023.