Domain Review

Focused crawling: a new approach to topic-specific Web resource discovery[1]

Context The paper presents a Focused Crawler, a specialized tool designed to address the challenges posed by general-purpose web crawlers due to the rapid growth of the Web. Unlike traditional crawlers that attempt to collect all accessible documents on the Web, the focused crawler selectively searches for relevant pages for predefined topics, based on example documents rather than keywords. By avoiding areas of the Web that are not relevant and concentrating on important zones, the focused crawler significantly conserves hardware and network resources while maintaining better up-to-date results.

Components of the Focused Crawler Two main components guide the focused crawler:

- **Classifier**: It assesses the relevance of a document with respect to the subject.
  - Creation of a taxonomy from relevant example documents.
  - The classifier assigns a relevance score $R_c(q)$ to each document q: this is the relevance of the document concerning the subject c. There are two types of classifiers (Hard focus and Soft focus).
- **Distiller**: The distiller measures the centrality of explored pages to determine visit priorities. In other words, it identifies pages that contain a large number of links to relevant pages (called "hubs"). This helps guide the crawler toward the most influential pages in the context of the topics of interest. Here are its two main characteristics:
  - In a graph (representation of a network of web pages), edges represent links between pages (nodes). Traditionally, in HITS*, edges are considered to have a unit weight, meaning all connections are treated equally. However, for focused crawling, edges must have variable weights, as some relevant web pages may point to non-relevant pages, and conversely, some non-relevant pages may point to relevant pages. The main idea here is that it is necessary to treat asymmetrically how hubs and authorities are connected to avoid the propagation of prestige (or influence) to non-relevant pages. Two weight matrices are proposed to represent this asymmetry:
    - EF[u; v]: The weight of the edge from page u to page v is the probability that u linked to v because v is relevant to the subject. This prevents the transmission of influence from relevant hubs to non-relevant authorities.
    - EB[u; v]: The weight of the edge from page v to page u is the probability that v linked to u because u is relevant. This prevents a relevant authority from spreading its prestige to a non-relevant hub.
  - Additionally, a relevance threshold is used to filter authorities in the graph. This means that only pages with a minimum relevance for the subject will be included in the set of authorities.

  The operation of the distiller is as follows:
  - Construct the set of edges E, keeping only links between pages on different sites, and applying the weights of the forward (EF) and backward (EB) edges.

– Execute iterations of the algorithm using these weighted edges while restricting the set of authorities based on the relevance threshold.

* HITS Algorithm: The HITS algorithm, proposed by Kleinberg in 1999, is used to identify important web pages based on their incoming and outgoing links. It assigns two types of scores to each page: the **hub** score and the **authority** score.

- **Hub**: A page that contains many links to other pages considered as authorities.
- **Authority**: A page that is linked by many hubs.

The HITS algorithm works as follows: for each page v in the graph,

$$a(v) = \sum_{u \in \text{Hubs}(v)} h(u)$$
$$h(v) = \sum_{w \in \text{Authorities}(v)} a(w)$$

where Hubs(v) denotes the set of pages pointing to v, and Authorities(v) denotes the set of pages pointed to by v.

- Pages with high authority scores are considered good sources of information on a given topic.
- Pages with high hub scores are useful for discovering other relevant pages.

Operation of the Focused Crawler

- Page exploration: As the crawler explores the Web, it collects data about the visited pages, including their outgoing links (to other pages). The pages are evaluated by the classifier, which assigns them a relevance value (R) based on their content in relation to the crawler's topics of interest. The most relevant pages are prioritized for exploration.
- Link storage: The crawler maintains a link graph on disk. This graph contains two types of link lists for each page: outgoing links (forward edges), which are pages that a given page points to, and incoming links (backward edges), which are pages that point to this page.
- Periodic execution of the distiller: At regular intervals, the crawler temporarily halts, and the distiller is executed. The distiller analyzes the link graph to identify hubs (pages with many links to relevant pages) based on the number of outgoing links pointing to pages evaluated as relevant (having a high relevance score). After identifying the most central and relevant hubs, the distiller updates the crawler's priorities by indicating which pages to revisit (the hubs), as well as the new unvisited pages cited by these hubs. These pages then become priority candidates for the crawler's next exploration steps.

The process is iterative: the crawler explores new pages, integrates the results into its link graph, and the distiller is re-executed at regular intervals to adjust the exploration strategy. The distiller optimizes exploration by redirecting the crawler toward areas of the Web where it is more likely to find relevant content.
Before distillation: The crawler explores pages based on a simple priority queue, relying on the relevance score R assigned by the classifier.
After distillation: The distiller reorganizes this queue based on the results obtained. Pages identified as hubs, and those directly linked to relevant hubs, are prioritized for visitation.

Evaluation

- Rate of relevant page acquisition (Harvest Rate): The absolute acquisition of pages for different topics was measured. This aimed to verify whether the crawler could capture enough relevant pages to justify its use in a focused framework. The same classifier used for the focused crawler was employed.
- Coverage (recall): Coverage measures how many relevant pages a crawler can discover. However, recall is difficult to measure for a focused crawler due to the complex and subjective nature of topics of interest on the Web. Solution: Disjoint fractions of the seed URLs were used to launch distinct crawls. By comparing the rate of relevant page acquisition between these crawls, the researchers could assess whether the crawler consistently discovered relevant pages from different starting points. If the crawls retrieve relevant pages at similar rates, it suggests that the crawler is robust and well covers the topic.
- Quality of resource discovery: This indicator is based on concrete examples (anecdotes) that demonstrate that the crawler effectively discovers high-quality relevant resources on specific topics.

A Personalized Machine-Learning-Enabled Method for Efficient Research in Ethnopharmacology: The Case of the Southern Balkans and the Coastal Zone of Asia Minor. [2]

Context Experts in ethnopharmacology face several challenges when it comes to identifying and retrieving documents and resources related to their field of study. These challenges include the large volume of sources to monitor, the variety of formats used, and the varying quality of language across different sources. This phenomenon is tied to the "big data" problem in analyzing such information. This study seeks to understand if and how experts can be effectively supported by AI tools in the task of researching ethnopharmacological literature.

Methodology Database : 50 documents that are highly relevant to the study topic, chosen by experts, constitute the seeds. Based on these 50 documents, a search was conducted to identify other relevant documents (extraction of references from these documents). This step led to the identification of a total of 427 additional documents. To expand the corpus without prior knowledge of relevant documents, a crawling method was used. This method involved randomly following the references in the visited publications through uniform sampling. By following this approach, 800 additional publications were retrieved. Before finalizing the corpus, a cleaning process was carried out to remove duplicates among the extracted documents. The final corpus reached a total of 1,012 documents in addition to the initial 50 seeds.
Components of the created focused crawler :

- Classifier: classifies relevant documents
  - AL (Active Learning)
  - LSTM-based
  - Training and evaluation: pre-trained on 50 documents (27 relevant and 23 irrelevant), followed by an evaluation of the active learning process:
    - Test sample of 100 documents, of which 50 are relevant and 50 are irrelevant
    - 4-fold cross-validation

        * A budget of 250 queries available to the user; the model asks the user to classify documents for which it is the least certain (with a classification probability around 0.5), and the model weights are updated (every $k = 10$ documents)

- RL algorithm:
  - DQN Network (Q-Network and target-Q-Network), utilizing an ϵ-greedy policy (to decide between exploitation and exploration)
  - Learns to visit URLs of relevant documents (reward of 1 when a relevant document is visited, otherwise 0)
    $\rightarrow$ Document relevance is determined by the AL classifier, and then it is checked for selected keywords
  - Frontier: stores URLs to visit (each time a document is visited, its references are stored in the frontier); Closure: stores already visited URLs (to avoid revisiting them)
  - State: frontier and closure
    Action: visit a URL from the frontier
  - The process starts with a seed: 50 highly relevant documents chosen by experts
  - Evaluation Metric : Harvest Rate
    Harvest Rate $= \frac{\text{number of relevant document fetched}}{\text{total number of fetched documents}}$

Results The results show that AI-based tools improved the efficiency and productivity of experts in ethnopharmacology. In the experiment, the algorithm retrieved 420 relevant documents in just 7 hours, compared to an estimated 36 hours if a human expert had performed this task manually.

Tree-based Focused Web Crawling with Reinforcement Learning[3]

Methodology and Main Components

- **Keyword Set Expansion**: Keywords and seeds often represent the crawler's only prior knowledge of a given subject $\rightarrow$ they determine which pages the crawler will explore $\rightarrow$ an irrelevant set of keywords can lead the crawler to follow URLs that diverge from the topic of interest.
  - An initial keyword set (KS) is provided, containing only keywords highly related to the target topic (C).
  - The paper proposes an expansion method that discovers new keywords from a textual document corpus (Dtr). All words in the texts from Dtr are considered as potential candidates for new keywords.
  - Each word is represented as a vector using word2vec. To determine if a candidate word (w) is a new keyword, a semantic score (CS(w)) is measured, which is the average cosine similarity of this word with each existing word in KS. If this score exceeds a given threshold (b), the word is considered a keyword and added to a new set (K0).
  - This distinguishes it from methods like TF-IDF or TextRank, which are less selective and may lead to the retrieval of irrelevant keywords.
- **Reinforcement Learning (RL) model** whose reward function is learned by a deep learning model called KwBiLSTM (a Bidirectional Long Short-Term Memory model). This model takes as input texts in the form of matrices containing words vectorized by word2vec and the words from the KeyWord set.
  The RL model is represented by an MDP formulation (used to model decision-making problems

in stochastic environments), and the model used is the Double Deep Q-Network (DDQN).
State Representation: Information about the path followed up to the URL u(t) (series of visited pages before reaching the current URL) is aggregated, and then 3 pieces of information are used:

  - The reward received at time t, which reflects the evaluation of the visited URL.
  - The inverse of the distance between URL u(t) and the closest relevant node in the followed path. This measures the URL's proximity to a relevant content source.
  - The relevance ratio of the path, which indicates how much the followed path contains relevant pages.

  Actions are based on:

  - The existence of keywords in the URL text.
  - The existence of keywords in the anchor text of the source page.
  - An estimate of the likelihood of its relevance provided by KwBiLSTM, based on the web page's title.

  Use of the "hubs" principle: → Idea: A web page belonging to a relevant website is likely to lead to other relevant sites, even if this particular page is not relevant. Web pages with this property are called "hubs." Hub score is given by two values:

  - The ratio of relevant web pages from the URL domain found up to step t.
  - The relevance of the unknown domain, which is set to 0.5 if the domain has not been visited before, otherwise it is set to 1.

  (I did not understand how the 2 hub features are used)

- **Tree-Frontier**: The Tree-Frontier algorithm serves to optimize the process of updating and selecting actions in a web crawler based on reinforcement learning. By representing the frontier of URLs to explore in the form of a decision tree, this algorithm reduces the time complexity of synchronized updates and improves sampling efficiency. Using experience samples to guide the partitioning of the tree nodes, Tree-Frontier facilitates a more accurate selection of actions by focusing only on representative samples, which helps quickly identify the most relevant web pages to explore.

Evaluation

- data: Real-world data from several topics belonging to the OpenDirectory project (Dmoz): indexing around five million URLs covering a wide range of topics. The training set of irrelevant web pages consists of URLs from general topics (e.g., sports). In cases where the relevant subject, C, is one of the aforementioned super-topics (e.g., sports), the web pages are removed from the training set of irrelevant web pages. On the other hand, if the relevant subject, C, is a subcategory of one of these super-topics, there is only a low chance that the web pages from the super-topics belong to the target subject.
  seed: Evaluations are made starting from a single seed (URL), unlike common experimental settings (but less realistic) that use a large number of seeds. Then, the average results from 10 different exploration runs starting from a single seed are used. Seeds are selected using Google search and are not connected to each other.

- metrics:

  - ability to find new keywords: percentage of new words found compared to the number of keywords initially provided.

  - classification performance: recall, precision.
  - overall performance of the focused crawling system: Harvest Rate and number of unique domains crawled (where a domain is considered relevant if it contains at least one relevant page. This emphasizes the crawler's ability to discover diverse content).
- comparisons with state-of-the-art indexing robots and simpler ones.

Results

- expansion: successfully discovers new keywords.
- classification: KwBiLSTM outperforms the two baseline methods, ACHE's SVM and Plain BiLSTM.
- focused crawler: TRES surpasses other methods in terms of Harvest Rate. The fewer maximum pages retrieved per domain, the more TRES discovers relevant domains while maintaining a good harvest rate.

Bootstrapping Domain-Specific Content Discovery on the Web[4]

Context The design of DISCO is inspired by the iterative process that domain experts typically follow to discover domain-specific websites:

- Submit queries to a search engine.
- Manually evaluate search results based on their expertise and knowledge of the domain.
- Use the newly acquired information to refine and reformulate queries.

In DISCO, the goal is to replace this manual evaluation with an automated system that uses a web ranking component while automating searches via search engine APIs.

2 Types of Methods for Discovering Web Pages

- **Search-based**:
  - Keyword search: involves submitting specific words or phrases to search engines to obtain a list of results containing web pages potentially relevant to a given domain.
  - Related search: the search engine analyzes a reference website (the one selected as a starting point) and attempts to find other sites that share common characteristics, such as the topics covered, the keywords used, or the types of products or services offered. (the search engine algorithm takes care of it)

  Existing approaches show certain limitations, including dependence on a precise classifier and limited adaptability to other domains. Iterative searching with domain-specific keywords can be effective in some cases, but it is not always applicable to all types of content discovery.
- **Crawling-based**:
  - Forward crawling: extracts links from already discovered web pages and follows them to discover new pages within the same area of interest. Based on 2 classifiers: 1. Critic (categorizes pages as relevant or not for the targeted domain); 2. Apprentice (learns in real-time to identify the most promising links to follow within pages relevant to the domain).
  - Backward crawling: discovers new pages by exploring the backlinks of already known pages (requires going through search engine APIs).

DISCO Architecture

- **Website Discovery Component:** This component takes as input a list of already discovered websites and returns a new list of discovered websites after performing search operations. There are 4 possible discovery operators:
  - Forward crawling
  - Backward crawling
  - Keyword search
  - Search for similar sites

  → Multi-armed bandit strategy: allows for selecting the best search operation at each iteration.

  Keyword Search Process: *The goal is to use a search engine to find websites relevant to a specific domain (e.g., weapon forums). The first step is to collect initial web pages (or "seeds"). These pages serve as entry points for exploring other related pages. From these pages, we begin by extracting domain-specific keywords (selecting the most frequent and relevant terms for the specific domain in question). These keywords are then used to submit queries to a search engine. However, keywords extracted from the body text of web pages are often "noisy," meaning they may include irrelevant terms. Keywords extracted from metadata tags (such as the <meta name="description"> and <meta name="keywords"> tags in HTML) are generally more precise and domain-specific. Nonetheless, these keywords (such as pistol, sale, Texas) can be too general. They often lack context, and when submitted to the search engine, they may return overly broad or irrelevant results. This is where query expansion comes into play: the keywords are combined with base terms (or "seed terms") that provide more context about the domain. For example, for weapon forums, a good base term could be "weapon forum." By combining this with the keywords, you get more specific queries, such as "weapon forum pistol," "weapon forum sale," and "weapon forum Texas". This process helps refine searches and ensures the results are more relevant to the targeted domain.*

- **Website Ranking Component:** Many focused crawling techniques depend on an accurate classifier for a specific domain, which can be a barrier. Indeed, in many cases, experts have access to only a small set of relevant sites at the beginning of the process, and this limited sample is insufficient to train an effective classifier. If the classifier is weak or inaccurate, it gradually decreases the efficiency of discovering new relevant sites.
  For DISCO: since there is no pre-established model to recognize relevant sites for a domain, the component ranks each site based on its similarity to the starting sites. This similarity can be based on shared characteristics among sites, such as content, link structure, or other metrics. It produces an ordered list of sites, allowing determination of which are most likely to be relevant to the domain of interest.

  Ranking functions:
  - Ranking based on regression (binomial, SVM)
  - Similarity (Jaccard, cosine)

  DISCO utilizes a combination of these ranking functions.

Evaluation

- Data: various domains
  Irrelevant sites: The ranking evaluation was performed by randomly selecting 10,000 sites from the DMOZ directory as negative candidates. Given that the studied domains are rare and that DMOZ contains approximately 4 million sites, the probability that the selected examples are relevant to the domain is low.

- Metrics:
  - Ranking: P@k, Mean Rank (better performance for the ensemble strategy than using the functions individually)
  - Discovery: Harvest Rank (compares DISCO to state-of-the-art strategies like SEEDFinder: 300% higher HV compared to other baselines), Coverage (To evaluate coverage, the union of sites discovered by all discovery methods is used as an approximation of the ground truth. The BANDIT strategy consistently achieves the largest intersection among all methods, suggesting that each operator explores a different region of the web, allowing the BANDIT strategy to cover a broader portion.)

Finding Seeds to Bootstrap Focused Crawlers[5]

Importance of "Seeds" The effectiveness of a focused crawler heavily depends on the quality and number of seeds provided.

- **Properties of t-graphs**
  A t-graph is a subgraph of the Web where nodes represent pages on a specific topic, and edges represent links between those pages. Several properties of these t-graphs influence crawler performance:
  - The number of outgoing links in a t-graph follows a Zipf distribution, meaning most pages have few links to other pages on the same topic.
  - These components are often connected via off-topic pages. Consequently, to move from one component to another, a crawler may need to traverse many non-relevant pages, which increases the time to collect relevant pages. For example, a crawler starting in component A might fail to reach a distant component, such as component C, if the path is too long or contains too many off-topic pages.

  Increasing the number of seeds distributed across different components can improve the t-graph coverage (i.e., the number of relevant pages collected) and the crawler's efficiency. For instance, placing a seed in component C enables the crawler to more easily collect pages within that component without traversing numerous non-relevant pages.
- **Limitations of Web Taxonomies for Seed Selection**
  Earlier approaches used Web taxonomies as sources to obtain seeds. However, these taxonomies have limitations:
  - If a topic is underrepresented or emerging (e.g., H1N1 information), there may not be enough pages to effectively guide the crawler.
  - Pages on a given topic may partially belong to multiple categories, making it difficult to precisely select seeds from a taxonomy. In such cases, users might need to manually adjust or modify the taxonomy to meet their needs.

Seed Finder Framework The idea is that by issuing queries related to a topic, it is possible to retrieve relevant pages that can serve as seeds. A pseudo-relevance feedback strategy is employed: an iterative process where a search engine uses the results of an initial query to refine subsequent search results, assuming that the top-ranked documents are relevant to the original query.

- **Challenges of the Approach:**

  - The terms describing the topic of interest and composing the queries are not fully known in advance.
  - Solution: An iterative pseudo-relevance feedback process is used to continuously generate new queries based on terms extracted from documents retrieved in the previous query.
  - Start with a simple initial query containing a few terms directly related to the topic. Based on the retrieved documents, new terms are selected to compose a new query.
  - Document relevance judgments for seed selection are performed by the search engine's ranking algorithm. While many documents may be retrieved for a query, not all are suitable as seeds. In most ranking methods, precision decreases as recall increases.
  - A topic filter classifier is used to filter retrieved documents, selecting those most likely related to the topic and suitable as seeds.
  - Short query submissions: Submitting several short queries rather than a single long one is preferable. For example, submitting 10 short queries with relevant terms and retrieving 100 results for each query is likely to lead to more relevant pages than submitting one long query and retrieving 1,000 results.

- **Exploration and Exploitation:** The goal is to achieve the broadest coverage of relevant pages while minimizing the number of queries submitted. The query submission policy must balance two factors: exploitation (choosing the best actions based on known information) and exploration (investigating actions that might seem suboptimal in the short term but could improve results in the future, ensuring diversity in the set of generated seeds).

- **Algorithm:**

  - Input: The algorithm takes an initial query as input.
  - Iterative Query Construction: Iteratively produces queries using terms selected from the response pages.
    * The `ProcessQuery` function retrieves two sets of documents: those considered relevant (D^+) and those deemed non-relevant (D^-), using a topic filter.
    * The URLs of relevant documents are added to the set of selected seeds.
    * A new query is generated using the `BuildNextQuery` procedure, which takes the current query and the sets D^+ and D^- as inputs, leveraging exploration and exploitation strategies.

    These steps repeat until a stopping criterion is met, such as obtaining a predefined number of seeds.

Evaluation

- **Parameters:**

  - Four topics of varying rarity were tested.
  - SeedFinder was set to stop after collecting 10,000 URLs from search engine responses.
  - For `ProcessQuery`, $K = 50$, corresponding to the maximum number of responses returned per query by Bing.
  - Topic filters were implemented using SVM classifiers trained with 30 positive and 30 negative examples for each topic.
  - To evaluate the effect of seeds on crawling, several configurations were compared: manually chosen seeds, seeds provided by DMOZ, and URLs resulting from submitting the initial query to the search engine.

- **Metrics:**
  - Coverage: An approximation of exhaustive crawling, calculated as the union of graphs obtained with all crawling configurations.
  - Seed Quality: The percentage of seed pages considered relevant for each topic. A good seed should lead to relevant pages, even if the seed itself is not relevant.
  - Seed Diversity: The distribution of hosts to which the seeds belong.
  - Harvest Rate: The efficiency of collecting relevant pages.
- **Results:** More seeds result in better coverage and higher harvest rates.

Query Expansion by Prompting Large Language Models[6]

Context Query expansion is a crucial technique for improving recall in information retrieval systems. This method involves enriching an initial query with additional terms to retrieve relevant documents that may not share lexical overlap with the original query. Traditional methods, such as Pseudo-Relevance Feedback (PRF), rely on the assumption that the initially retrieved documents are relevant, which is not always the case, especially for short or ambiguous queries.

The paper proposes a new approach to query expansion using Large Language Models (LLMs), which leverage their generative capabilities. The authors examine various types of prompts, such as zero-shot, few-shot, and Chain-of-Thought (CoT), demonstrating that CoT prompts are particularly effective.

Evaluation

- **Metrics**: Recall@1K, MRR@10, and NDCG@10, calculated using BM25.
- **Datasets**: MS-MARCO and BEIR.
- **Results**:
  - PRF approaches remain effective, especially for specialized datasets.
  - LLMs appear particularly beneficial for question-answering datasets, where they generate relevant answers that help retrieve pertinent passages.
  - CoT prompts (which break down the model's answer into steps) seem to particularly enhance performance.
  - By integrating PRF documents into prompts, it is possible to improve results while maintaining robustness for smaller model sizes.
  - Larger models (with more parameters) generally yield better performance.
- **Limitations**:
  - Analyzing prompts in a dense retrieval context could be a future research direction: the study focuses solely on sparse retrieval systems (BM25), leaving aside dense retrieval systems, which might benefit less from query expansion.
  - The use of specific language models (e.g., Flan); how do the results generalize to other models?
  - Exploring model distillation to reduce model size while retaining their effectiveness.

LLMs in the Loop: Leveraging Large Language Model Annotations for Active Learning in Low-Resource Languages [7]

- A good example of LLM use for Active Learning (AL).
- Use case: Named Entity Recognition (NER) in rare languages.
- Classification model used: AfroXLMR-mini, a compact multilingual language model based on the BERT architecture, fine-tuned specifically for African languages.
- Comparison of several LLMs: GPT-4, Mistral 7B, Llama2.
- Prompt strategy: "We employ a few-shot prompting approach [6] to extract entity labels from large language models. Our prompt template includes placeholders for the language, annotation examples, and sample tokens. The prompt directs the LLM to label each token, providing examples in the target language. It specifies the desired output format and set of labels. A zero-shot approach might not produce the desired format, leading to additional post-processing. Therefore, we opted for a few-shot approach, providing examples and instructions to ensure the model outputs match the required format."
- Results:
  - "GPT-4-Turbo demonstrated consistent and accurate annotations, while others showed strengths in specific contexts. This emphasizes the necessity of selecting appropriate models based on task requirements."
  - "Potential cost savings of at least 42.45 times compared to human annotation."

What Makes Good In-Context Examples for GPT-3? [8]

Choose the m-nearest neighbors for ICL and GPT3

Query2doc: Query Expansion with Large Language Models [9]

ActiveLLM: Large Language Model-based active learning for textual few-shot scenarios [10]

Active learning involves optimizing the learning process by selecting the data to annotate. Unlike traditional methods, where a dataset is fully annotated before the model training, active learning allows the model to query an annotator (or oracle) for labels only for the examples it considers the most informative. This reduces the cost and effort of annotation while improving the model's performance by focusing on the instances that have the most impact on its learning.

A commonly used method is the uncertainty strategy: this method aims to select the most uncertain instances based on a classifier learned iteratively during annotation. There are also "model mismatch scenarios," in which the instance selection model (query model) differs from the model used for the final classification task (successor model).

Cold start: to know which data are the most interesting to be annotated, models need a sufficiently large amount of base annotated data. This problem often means that active learning does not improve the performance of the classifier.

This paper aims to show that this problem can be solved by using a large language model as a query model.

FreeAL: Towards Human-Free Active Learning in the Era of Large Language Models [11]

The FreeAL framework aims to implement active learning without any initial labeled data. Here's an overview of the methodology:

- Generating Pseudo-Labels : To construct a dataset denoted as D_{gen}, the LLM is prompted to generate text that mimics the style of the unlabeled dataset. Simultaneously, the LLM provides pseudo-labels for the generated samples.
- Training a Small Language Model (SLM) : An SLM is then trained using the pseudo-labels obtained from the LLM. This model serves to refine the labeling process.
- Few-Shot ICL with the LLM : For each unlabeled instance x_i, the m nearest neighbors are selected from D_{gen}. The LLM employs these neighbors to enhance its understanding and provide better annotations. (LLMs exhibit exceptional performance when utilized with few-shot in-context learning (ICL) prompts.)
- Filtering and Refining Labels : After labeling, the SLM identifies correctly labeled data, which contributes to a new dataset referred to as D_{demo}. It also flags samples with poor-quality labels using a filtering process based on loss functions to detect noise in the labels.
- Iterative Annotation Process : The process iteratively continues, where the LLM re-labels the poorly annotated data until convergence is achieved.

Enhancing Text Classification through LLM-Driven Active Learning and Human Annotation [12]

Selective Annotation Makes Language Models Better Few-Shot Learners [13]

new way of ICL to improve prompt

References

[1] B. Chakrabarti, S. ; van den Berg M.; Domc. Focused crawling: a new approach to topic-specific web resource discovery. *Computer Science and Engineering*, 1999.

[2] A.; Kalpoutzakis E.; Giannakopoulos G. Axiotis, E.; Kontogiannis. A personalized machine-learning-enabled method for efficient research in ethnopharmacology. the case of the southern balkans and the coastal zone of asia minor. *Applied Sciences*, 2021.

[3] Kelesis D. Pollatos V. Paliouras G. ; Kontogiannis, A. and G. Giannakopoulos. Tree-based focused web crawling with reinforcement learning. 2022.

[4] Freire J. Pham K., Santos A. Bootstrapping domain-specific content discovery on the web. 2019.

[5] Soares da Silva A. Freire J. Moura E. Vieira K., Barbosa L. Finding seeds to bootstrap focused crawlers. 2015.

[6] Qin Z. Wang X. Bendersky M. Jagerman R., Zhuang H. Query expansion by prompting large language models. 2023.

[7] Khodadadi M. Nurullah Gumus M. Granitzer M. Kholodna N., Julka S. Llms in the loop: Leveraging large language model annotations for active learning in low-resource languages. 2024.

[8] Zhang Y. Dolan B. Carin L. Chen W. Liu J., Shen D. What makes good in-context examples for gpt-3? 2021.

[9] Wei F. Wang L., Yang N. Query2doc: Query expansion with large language models. 2023.

[10] Reuter C. Bayer M. Activellm: Large language model-based active learning for textual few-shot scenarios. 2024.

[11] Zhao J. Wu R. Lin M Chen G. Wang H. Xiao R., Dong Y. Freeal: Towards human-free active learning in the era of large language models. 2023.

[12] Makrehchi M. Rouzegar H. Enhancing text classification through llm-driven active learning and human annotation. 2024.

[13] Henry Wu C. Shi W. WangW T. Xin J. ZhangF R. Ostendorf M. Zettlemoyer L. A. Smith N. Yu T. Su H., Kasai J. Selective annotation makes language models better few-shot learners. 2022.