Appendix A. Prompts

When constructing contrastive learning training data, we use GPT's classification to obtain L_{pos} and L_{neg}. For each entity in the top-T of L_0, determine whether its attributes are consistent with S^{pos} (S^{neg}). If it is consistent, GPT should output 1; otherwise, GPT should output 0. Next, these entities that GPT deems consistent with the attributes of S^{pos} (S^{neg}) will be merged with S^{pos} (S^{neg}) to form L_{pos} (L_{neg}). The prompt to classify entity is shown in Table 12.

In GenExpan, each round uses three example entities to construct $Prompt_g$, which is used to generate entities that are semantically similar to them. $Prompt_g$ is as shown in Table 13.

In the process of enhancing GenExpan with Chain of Thought prompting, we employ LLM to generate fine-grained class names. The prompt, $Prompt_c$, is shown in Table 14.

Appendix B. Implementation Details of RetExpan

In RetExpan, sentences are tokenized using the WordPiece tokenizer and then fed into a 12-layer Transformer initialized with $BERT_{BASE}$ weights. To optimize training efficiency and preserve semantic knowledge learned by BERT, we freeze the first 11 layers of the encode, thus only the last layer is fine-tuned. When tokenizing, to ensure that mentions of entities in long sentences are not truncated, we have implemented entity focus, guaranteeing the presence of entities within the sentence. During expansion, we use one-shot expansion to directly obtain the preliminary expansion results L_0. In RetExpan with contrastive learning strategy, each epoch during training alternates between computing and optimizing entity prediction loss and contrastive learning loss.

To ensure that the encoder acquires knowledge from the corpus, we trained it for 20 epochs on 8 RTX 3090 GPUs. The hyperparameters for training learning rate, batch size, weight decay, label smoothing factor η were set to 4e-5, 128, 1e-2, and 0.075 respectively.

Appendix C. Implementation Details of GenExpan

In GenExpan, we use LLaMA-7b as the base model. Initially, we train for 1 epoch on the corpus using 6 A100 GPUs. The training hyperparameters learning rate, batch size, gradient accumulation steps, weight decay, gradient clipping are set to 1e-5, 4, 8, 1e-4, and 1.0 respectively.

During entity generation in GenExpan, we utilize prefix-constrained beam search with a beam size of 40 to generate 40 entities at a round. For entity selection, we compute the positive similarity score for each entity and select those whose scores are in the top 0.7 as the results for the current round. When no new entity is generated for 20 consecutive rounds, the generation process will end and move to re-ranking.

TABLE 10. TYPES OF ULTRA-FINE-GRAINED SEMANTIC CLASSES. CLS.: SEMANTIC CLASS

$\lvert\mathcal{A}^{pos}\rvert$	$\lvert\mathcal{A}^{neg}\rvert$	#Ultra-fine-grained CLS.
1	1	238
1	2	5
2	1	9
2	2	7
3	3	2

Appendix D. Details of UltraWiki

UltraWiki contains 10 fine-grained semantic classes, which fall under 5 coarse-grained semantic categories: Organization, Location, Product, Person, and Miscellaneous. The number of entities in each fine-grained semantic class ranges from 45 to 952. Each fine-grained semantic class has 2 to 3 attributes, and depending on the combination of attributes, several ultra-fine-grained semantic classes can be derived from a single fine-grained class. Detailed data is displayed in Table 11.

The number of combinations from positive and negative attributes varies significantly, leading to substantial differences in the number of ultra-fine-grained semantic classes. The count of UltraWiki's positive and negative attributes and their corresponding ultra-fine-grained classes are presented in Table 10.

Appendix E. Discussion of Generalization of Dataset Construction Pipeline

UltraWiki is built through a multi-step pipeline involving:

(1) **Identifying fine-grained semantic classes and entities**
(2) **Extracting entity-relevant sentences**
(3) **Annotating entity attributes**
(4) **Generating ultra-fine-grained semantic classes** via positive and negative attribute constraints

While our pipeline utilizes Wikipedia's structured data (e.g., entity lists, hyperlinks, and Wikidata properties), its core principle, leveraging attribute constraints to define ultra-fine-grained entity sets, is domain-agnostic. The key steps (finding entities, tagging them in context, characterizing their attributes, and forming semantic classes) can be implemented using alternative techniques to replace Wikipedia's built-in structure. For instance, in biomedicine, a similar Ultra-ESE dataset can roughly be constructed as follows:

TABLE 11. FINE-GRAINED SEMANTIC CLASSES DETAIL. CLS.: SEMANTIC CLASS

Coarse CLS.	Fine-grained CLS.	#Entities	#Ultra-fine-grained CLS.	Attributes
Organization	Canada universities	99	10	<Loc-Province>, <Type>
Location	China cities	675	50	<Province>, <Prefecture>
	Countries	190	68	<Continent>, <Driving-Side>, <Per-Capita-Income>
	US airports	370	74	<Role>, <Loc-State>
	US national monuments	112	12	<Loc-State>, <Agency>
Product	Mobile phone brands	159	7	<Loc-Continent>, <Status>
	Percussion instruments	128	10	<Type>, <Source-Continent>
Person	Nobel laureates	952	11	<Prize>, <Gender>
	US presidents	45	5	<Party>, <Birth-State>
Miscellaneous	Chemical elements	118	14	<Period>, <Phase-at-R.T.>

TABLE 12. THE PROMPT USED TO SELECT THE ENTITIES THAT ARE CONSISTENT WITH THE ATTRIBUTES OF S^{pos} (S^{neg}) IN THE TOP-T ENTITIES OF L_0 TO CONSTRUCT L_{pos} (L_{neg}).

I have a task that involves classifying candidate entities based on their alignment with a seed entity set. The seed entities are grouped together because they share certain attributes, referred to as seed attributes. I will provide a list of seed entities along with their seed attributes. Additionally, I have a list of candidate entities that are similar to the seed entities but may not necessarily share the same seed attributes. I need you to identify the seed attributes and use them to classify each candidate entity into one of two categories: 1) consistent with the seed entity set in terms of seed attributes, or 0) inconsistent with the seed entity set in terms of seed attributes. For the given N candidate entities, please output N values, each being 1 or 0, indicating whether each candidate is consistent (1) or inconsistent (0) with the seed entity set based on the seed attributes.

Input:
Seed entities: [Mark Twain, Ernest Hemingway, F. Scott Fitzgerald]
Candidate entities: [J.K. Rowling, Stephen King, Agatha Christie, John Steinbeck, Harper Lee, Charles Dickens, Virginia Woolf], total 7 entities
Output:
"result": [0,1,0,1,1,0,0]

Input:
Seed entities: [Golden Retriever, German Shepherd, Labrador Retriever]
Candidate entities: [Bengal Tiger, Beagle, Siberian Husky, African Elephant, Pug], total 5 entities
Output:
"result": [0,1,1,0,1]

Input:
Seed entities: [{Entity1}, {Entity2}, {Entity3}]
Candidate entities: [{Entity1'}, {Entity2'}, {Entity3'}, ...], total {} entities
Output:

TABLE 13. THE PROMPT USED TO GENERATE ENTITIES THAT ARE SEMANTICALLY SIMILAR TO 3 GIVEN ENTITIES.

iron, copper, aluminum and zinc.
math, physics, chemistry and biology.
{Entity1}, {Entity2}, {Entity3} and _____

(1) **Identifying Fine-Grained Entity Categories**

- *Domain-Specific Resources:* Ontologies like MeSH (Medical Subject Headings)[2] or Gene Ontology (GO)[3] provide structured biomedical concepts, serving as alternatives to Wikipedia for obtaining fine-grained categories and entity lists.

(2) **Collecting Entity Contexts**

- *Scientific Literature:* Context sentences can be extracted from PubMed[4] abstracts or full-text articles. Tools like GNormPlus [44, 45] can identify and normalize gene/protein mentions to ensure precise entity linking.

(3) **Annotating Entity Attributes**

- *Structured Databases:* Domain-specific resources like UniProt[5] can offer detailed annotations for protein properties (e.g., protein function, subcellular

2. https://www.ncbi.nlm.nih.gov/mesh/
3. https://geneontology.org/
4. https://pubmed.ncbi.nlm.nih.gov/
5. https://www.uniprot.org/

TABLE 14. THE PROMPT USED TO GENERATE A CLASS NAME THAT COVERS THE GIVEN ENTITIES.

Generate a class name that accurately represents the following entities. This class name should encompass all the given entities and reflect their shared characteristics. Examples: [Tiger, Lion, Cheetah] → Big Cats [Shakespeare, Tolstoy, Hemingway] → Famous Authors [Mercury, Venus, Mars] → Planets in the Solar System [{Entity1}, {Entity2}, {Entity3}] → _____

location). Manual curation can also be introduced to ensure benchmark quality.

(4) **Defining Ultra-Fine-Grained Classes**

- *Attribute Constraints:* Formulate ultra-fine classes by combining positive and negative attribute constraints. Our negative-aware semantic class generation algorithm can be directly applied at this step.

These steps can be adapted to different domains, while the essential goal is still to construct ultra-fine-grained semantic classes using attribute constraints. Certainly, manual curation remains inevitable; however, the rise of LLMs like ChatGPT can reduce human effort. Exploring how LLMs can facilitate high-quality, automated dataset creation remains a promising research direction.

Appendix F. Parameter Analysis

To explore the sensitivity of each key hyperparameter in RetExpan and GenExpan, we conducted a comprehensive analysis of the smoothing factor η and the re-ranking segment length l in RetExpan, as well as the number of positive and negative entities mined in the contrastive learning strategy $|L_{pos}|$ and $|L_{neg}|$, the Top-p of entity selection module in GenExpan, and the re-ranking segment length l, respectively. The results are depicted in Figure 7, from which we can observe that:

(1) The label smoothing η serves to mitigate the penalty for entities possessing similar semantics to ground-truth entities. In Ultra-ESE, there are fewer entities exhibiting similarity to ground-truth entities at the ultra-fine-grained semantic level, thereby diminishing the utility of label smoothing. Consequently, the fluctuation in model performance due to parameter variations is relatively not significant.

(2) Excessive segment length l in both RetExpan and GenExpan equates to degrading the segmented re-ranking strategy to a naive re-ranking approach, thereby introducing noise from irrelevant entities, as mentioned in the main text. Hence, there is a general decrease in both PosMAP@K as the segment length increases, with larger K values leading to greater declines.

(3) Regarding the ultra-fine-grained contrastive learning strategy in RetExpan, we observe that mining too many or too few positive and negative entities using GPT-4 results in performance degradation. Insufficient positive and negative entities fail to fully stimulate the potential of contrast

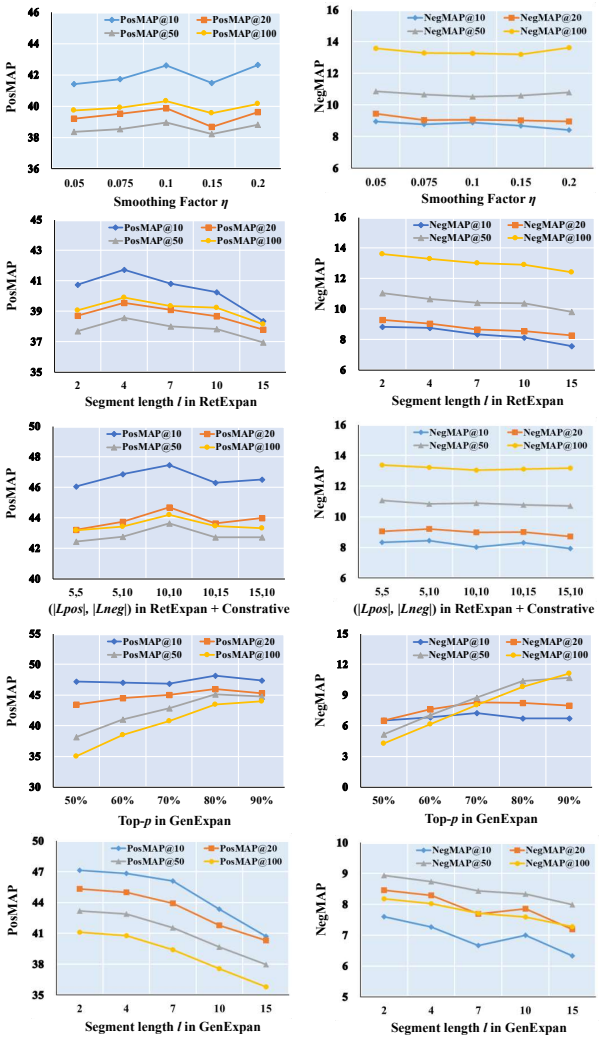

Figure 7. Parameter analysis experiments.

learning, whereas an excess of such entities introduces noisy entities. Therefore, as the number of entities to be mined increases gradually, PosMAP initially rises and then falls, and NegMAP exhibits a slight initial decline followed by a rise. However, overall, these two parameters demonstrate robustness, with even the worst-performing set of values outperforming the original RetExpan.

(4) A trade-off is also necessary for the Top-p. A too

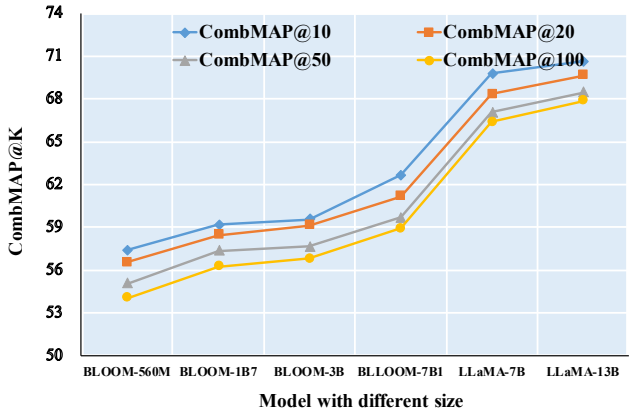

Figure 8. Comparison experiments on different LLM families and sizes.

small Top-p may fail to ensure the diversity of newly generated entities, while a too large Top-p may introduce an excessive number of irrelevant entities during iterative expansion. Consequently, we observe a decrease in PosMAP and a notable increase in NegMAP when Top-p is increased from 80% to 90%, which indicates that there is considerable space for improving the current rejection strategy for negatively targeted entities.

Appendix G. Analysis of Model Size

We utilize various families and sizes of LLMs as backbone model of GenExpan, including 560M, 1B7, 3B, 7B1 for BLOOM, as well as 7B and 13B for LLaMA. The experiment results depicted in Figure 8 align well with the expectation. Both the BLOOM and LLaMA families satisfy the scaling law: larger models tend to be more effective in Ultra-ESE, yielding better results. Moreover, LLaMA-7B outperforms BLOOM-7B1 at the same scale.

Appendix H. Impact of Negative Seeds on Performance

As shown in Tables 4 and 6 (in main body of paper), we have analyzed how the number of negative attributes affects model performance. Here, we further investigate how the number of negative seed entities impacts model performance, conducting experiments on the RetExpan model.

As shown in Figure 9, we progressively remove negative seed entities from the input. The results clearly indicate that as the number of negative seeds decreases, overall model performance declines. Specifically, PosMAP decreases, while NegMAP increases accordingly. Additionally, we observe that negative seeds have a greater impact on the Negative metric, and as the number of negative seeds increases, the performance gain gradually diminishes (diminishing marginal effect).

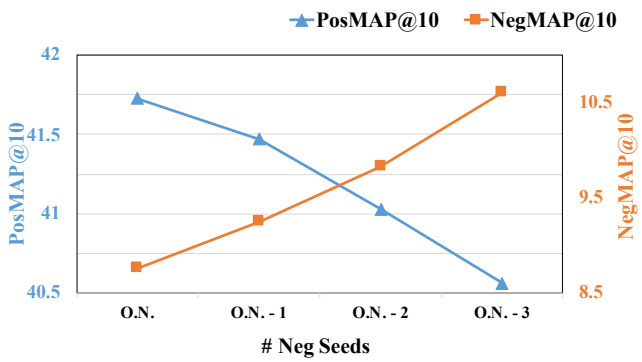

Figure 9. The figure shows the performance curve as the number of negative seeds varies. The x-axis, labeled "# Neg Seeds", represents the number of negative seed entities in input. O.N. means original number of negative seed entities.

Appendix I. Why GenExpan Omits Negative Seeds in Entity Generation

The primary reason for not using negative seeds during the entity generation stage is to maximize recall in this initial phase, ensuring the retrieval of as many entities belong to fine-grained class as possible. This intuition aligns with the first-stage retrieval (coarse ranking) in recommendation and retrieval systems.

In our preliminary experiments, we compared results with and without negative seeds, using the negative-seed-enhanced prompt described in Table 16. As reported in Table 15, we evaluated both the first-stage recall (coarse recall in entity generation) and end-to-end model performance (ComMAP). The results show that introducing negative seeds harms recall, which in turn leads to a decline in overall MAP performance.

TABLE 15. APPLYING NEGATIVE SEEDS TO THE ENTITY GENERATION PHASE (SIMPLY NOTED AS PHASE 1) OF GENEXPAN.

Method	Metric	@10	@20	@50	@100
GenExpan	Recall in Phase 1	93.58	92.43	74.14	52.30
	ComMAP	69.79	68.35	67.07	66.38
+ Neg. seeds in Phase 1	Recall in Phase 1	95.27	83.65	48.52	27.24
	ComMAP	65.91	63.79	61.05	60.02

Appendix J. Enhancing Performance of RetExpan and GenExpan Using Entity Descriptions

Leveraging seed entity descriptions can further enhance LLM's reflection on its own generation. On one hand, the contextual information of entities primarily comes from the provided corpus D, which may not be sufficient for capturing ultra-fine-grained semantic distinctions. Incorporating

TABLE 16. PROMPT USED IN THE GENERATION PHASE OF GENEXPAN WITH NEGATIVE SEEDS.

similar to iron, copper, aluminum, rather than wood, plastic, glass, it is zinc. similar to math, physics, chemistry, rather than history, art, music, it is biology. similar to {pos_ents}, rather than {neg_ents}, it is _____

TABLE 17. WE REPORT THE MAP AND P SCORES CORRESPONDING TO POS/NEG/COMB ON ULTRAWIKI. THE HIGHEST SCORE IS BOLDED.

Metric Type	Method Type	Method	MAP				P				Avg
			@10	@20	@50	@100	@10	@20	@50	@100	
Pos ↑	Retrieval Based (Ours)	RetExpan	41.73	39.53	38.55	39.91	54.58	58.03	66.76	77.23	52.04
		RetExpan + Contrast	47.45	44.68	43.63	**44.20**	59.83	62.02	**69.36**	**77.92**	**56.14**
		RetExpan + RA	44.74	42.23	40.66	40.27	58.31	61.36	66.58	73.97	53.52
	Generation Based (Ours)	GenExpan	46.79	45.00	42.89	40.80	59.77	62.15	66.26	66.57	53.78
		GenExpan + CoT	**50.39**	**47.80**	43.67	40.06	**62.74**	**64.45**	64.06	60.38	54.19
		GenExpan + RA	48.68	46.26	**43.80**	41.27	61.12	62.64	66.49	65.65	54.49
Neg ↓	Retrieval Based (Ours)	RetExpan	8.77	9.04	10.65	13.29	16.44	21.04	34.78	56.54	21.32
		RetExpan + Contrast	8.02	8.98	10.89	13.05	14.83	21.23	35.47	55.12	20.95
		RetExpan + RA	5.53	5.88	6.56	7.44	11.70	16.70	25.81	37.05	14.58
	Generation Based (Ours)	GenExpan	7.25	8.28	8.72	8.01	15.21	21.31	27.33	28.58	15.59
		GenExpan + CoT	7.79	9.29	8.15	6.89	15.97	22.66	23.52	21.90	14.52
		GenExpan + RA	7.16	8.54	8.63	7.70	14.79	21.18	26.67	26.84	15.19
Comb ↑	Retrieval Based (Ours)	RetExpan	66.48	65.25	63.95	63.31	69.07	68.50	65.99	60.34	65.36
		RetExpan + Contrast	69.72	67.85	66.37	65.57	72.50	70.39	66.95	61.40	67.59
		RetExpan + RA	69.60	68.18	67.05	66.42	73.31	**72.33**	**70.39**	68.46	69.47
	Generation Based (Ours)	GenExpan	69.77	68.36	67.08	66.40	72.28	70.42	69.47	68.99	69.10
		GenExpan + CoT	**71.30**	**69.25**	**67.76**	66.58	**73.39**	70.90	70.27	69.24	**69.84**
		GenExpan + RA	70.76	68.86	67.58	**66.78**	73.16	70.73	69.91	**69.41**	69.65

entity descriptions introduces additional external knowledge. On the other hand, entity-specific information can be diluted during training, especially for long-tail entities. Explicitly including entity descriptions in the prompt can help activate long-tail knowledge during inference.

To explore this, we propose a entity-based retrieval-augmented method (RA). Specifically, we retrieve entity introduction texts from Wikidata to enrich entity knowledge. During both training and inference, these entity descriptions serve as prefixes for all sentences, enhancing the model's understanding of ultra-fine-grained entity semantics. As shown in Table 17, this simple retrieval-augmentation strategy significantly improves performance for both RetExpan and GenExpan. Notably, for RetExpan, this strategy improves the comprehensive Comb metrics by an average of 4.11 points. The strategy primarily optimizes the Neg class metrics, leading to a significant reduction in the intrusion of negative target entities. However, its effect on Pos class metrics exhibits instability. For instance, the PosP@50 and PosP@100 of RetExpan+RA are instead lower compared to the original RetExpan. This phenomenon can be attributed to the fact that the extra introductions corresponding to each entity are currently fixed and do not adapt to changes in the entity context. Consequently, this increases the homogeneity of the entity embedding between irrelevant entities (neither positive nor negative target entities) and positive seed entities under the same fine-grained semantic class to some extent. Our point is further supported by the observation that RetExpan's MAP@100 for fine-grained semantic classes grows from 82.08 to 87.17 after the introduction of the retrieval augmentation strategy. This suggests that a dynamic and more fine-grained knowledge retrieval strategy needs further investigation.

Appendix K. Dynamic Segmentation

The re-ranking strategy uses the uniform segmentation method to reduce noise. To further enhance the model performance, we propose **Confidence-based Dynamic Segmentation (CDS)**, an algorithm that partitions a set of candidate indices T into segments based on a confidence threshold p. The candidates are ranked in descending order by their similarity sim^+ to a set of positive seeds P. As shown in Algorithm 1, the algorithm first computes the similarity scores sim^+ between candidates and positive seeds. It then iteratively normalizes these scores within the remaining candidate pool and accumulates them until the sum exceeds the threshold p, creating a segment. This process repeats until all candidates are segmented. The resulting segments reflect regions of high confidence in the candidate list, ensuring that each segment maintains a consistent level of similarity to the positive seeds.

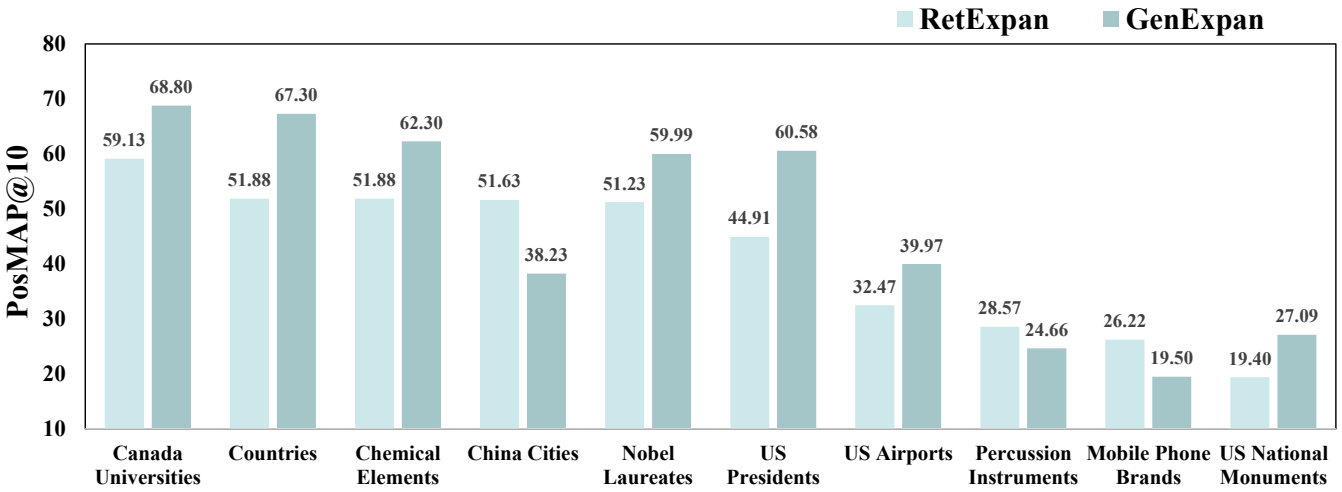

Figure 10. This figure shows the average PosMAP@10 of ultra-fine-grained semantic classes within each fine-grained class for RetExpan and GenExpan.

Algorithm 1 Confidence-based Dynamic Segmentation

Input:
  T: Candidate indices (sorted by sim^+ in descending order).
  P: Positive seed indices.
  p: Confidence threshold for segmentation ($0 < p \leq 1$).
Output: Segmented candidate indices.

Compute $sim^+ = \text{mean\_similarity}(T, P)$.
Initialize $i = 1$, segments $S = []$.
while $i \leq |T|$:
  Normalized scores $s_j \leftarrow \frac{sim_j^+}{\sum_{k=i}^{|T|} sim_k^+}$ for $j \in [i, |T|]$.
  Find the smallest $j > i$ such that $\sum_{k=i}^{j} s_k > p$.
  Add new segment $S \leftarrow S \cup \{T[i : j-1]\}$.
  Set $i = j + 1$.
return S.

TABLE 18. PERFORMANCE OF RETEXPAN AND THE RETEXPAN USING CONFIDENCE-BASED DYNAMIC SEGMENTATION, WHICH IS SIMPLY NOTED AS RETEXPAN (CDS).

Metric Type	Method	MAP				Avg
		@10	@20	@50	@100	
Pos	RetExpan	41.73	39.53	38.55	39.91	39.93
	RetExpan (CDS)	41.97	39.75	38.63	39.94	40.07
Neg	RetExpan	8.77	9.04	10.65	13.29	10.43
	RetExpan (CDS)	7.91	8.29	10.04	12.69	9.73
Comb	RetExpan	66.48	65.25	63.95	63.31	64.75
	RetExpan (CDS)	67.03	65.73	64.30	63.63	65.17

Appendix L. Challenges with Long-Tail / Fine-Grained Entities

For classes with high semantic overlap, we take **Non-Small Airports in Florida** and **Non-Large Airports in Florida** as an example. The target entity overlap between these two ultra-fine-grained classes reaches 63.64%. Even the best-performing GenExpan model struggles with this distinction, showing a performance drop of nearly 40 points compared to the average (69.8 → 30.3). Additionally, we observe entity "intrusion" between these two classes, where 9% and 13% of target entities that should belong exclusively to one class appear in the top-100 expanded entities of the other class.

TABLE 19. STATISTICAL TABLE OF THE AVERAGE NUMBER OF ACTUAL EXPANSIONS (EXPANDED) AND EXPECTED EXPANSIONS (EXPECTED) FOR SELECTED NON-LONG-TAIL ENTITIES AND LONG-TAIL ENTITIES.

Entity Type	Expectation	Expansion	
		not expanded	expanded
Non-long-tail	not expected	0.45	11.09
	expected	0.00	15.45
Long-tail	not expected	19.64	1.74
	expected	4.00	1.62

For long-tail entities, we report the PosMAP@10 performance for each fine-grained semantic class in Figure 10. From the figure, we observe that the model performs worse on semantic classes with a higher proportion of long-tail entities, which aligns with intuition. To illustrate this more clearly, we randomly selected 50 entities from the "US National Monuments" class, where 78% were long-tail entities and 22% were non-long-tail entities. Here, long-tail entities are defined as those with fewer than 1,000 words in their Wikipedia entries. Furthermore, we analyzed a confusion matrix of expansion results, categorizing entities into expanded, not expanded, expected to be expanded, and not expected to be expanded, as shown in Table 19. The results indicate that non-long-tail entities were expanded significantly more frequently than long-tail entities. Specifically, non-long-tail entities appeared 26.5 times on average in expansions, whereas long-tail entities had an expansion count of 3.4 under the same conditions.

This suggests that long-tail entities, due to their limited available information, are harder for the model to capture, leading to poorer expansion performance. This phenomenon further highlights the performance bottleneck for long-tail entities in fine-grained semantic classes, especially in cases like "US National Monuments", where long-tail entities dominate.

Appendix M. Ultra-fine-grained Semantic Classes Visualization

To better illustrate the distribution of entities across ultra-fine-grained semantic classes divided by different attributes, we have visualized them with t-SNE. In Figure 11, based on the semantic class of **Nobel Laureates**, we divided them into four semantic classes according to four attribute values [Physics, Chemistry, Physiology or Medicine, Literature] under the **Prize** attribute. Entities with different prizes show distinct distribution differences, and after refining entity representations through our ultra-fine-grained contrastive learning, the clustering effect becomes more optimal. In Figure 12, based on two attribute values each from the **Prize** and **Gender** attributes, we divided the entities into four semantic classes, each constrained by two attributes. From the figure, it can be observed that entities with the same prize tend to cluster better than those with the same gender. After refining representations through contrastive learning, entities with different genders can also be well distinguished, although the effect is not as good as in the case of single-attribute constraints. This also indicates that ultra-fine-grained semantic classification with multiple attribute constraints places high demands on the precision of semantic understanding, and that contrastive learning is highly effective.

Appendix N. Case Study

Figure 13 presents some case studies of GenExpan and the chain-of-thought reasoning technique. Overall, it is evident that GenExpan predominantly avoids expanding entities that do not belong to the same semantic class as the positive and negative seed entities. This emphasizes the relative tractability of traditional ESE tasks compared to the significant challenges posed by Ultra-ESE for existing LLMs, where the model cannot exclude the expansion of negative target entities and irrelevant entities (belonging to the same fine-grained semantic class).

For chain-of-thought reasoning, an interesting example is shown in Figure 13. We find that the ground-truth positive target semantics is "low-income countries", but the model erroneously infers the positive semantic class as "African countries", which encompasses relatively more low-income countries, and thus improves the recall of the positive target entities. Of course, some high-income African countries such as "Seychelles" are also incorrectly introduced. In

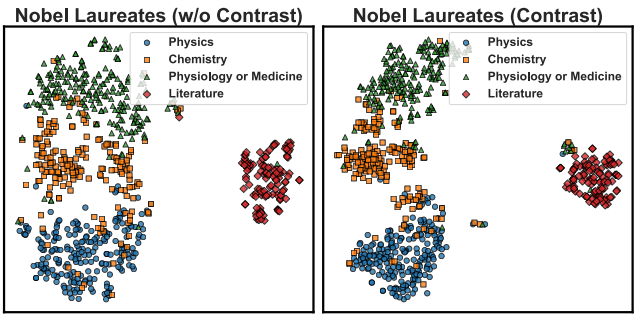

Figure 11. Comparison of clustering effects before and after contrastive learning on Nobel Laureates' **Prize** attribution (choosing 4 values [Physics, Chemistry, Physiology or Medicine, Literature]).

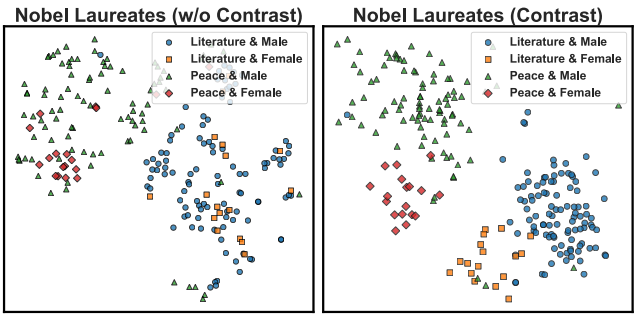

Figure 12. Comparison of clustering effects before and after contrastive learning on Nobel Laureates' **Prize** and **Gender** attributions ([Literature, Peace] $\times$ [Male, Female]).

fact, this error could have been avoided since the provided positive seed entity contains the non-African country "Saint Vincent and the Grenadines". This also shows the potential for improvement in the chain-of-thought reasoning strategy.

For retrieval augmentation proposed above, more positive target entities are introduced after fetching the additional knowledge. However, for some negative target entities that fulfill positive attributes, the retrieved knowledge may contain the introduction about its positive attributes, conversely misleading the model to overlook its negative attributes. For example, for the negative target entity "Xinxiang" in the second column, it also belongs to Henan province, i.e., it satisfies the requirement of positive attributes. Therefore, after concatenating the retrieved entity introduction "Xinxiang is a prefecture-level city *in northern Henan province*, China.", it is incorrectly introduced instead.

China Cities				Countries			
GenExpan		GenExpan + RA		GenExpan		GenExpan + CoT	
1	Changge +++	1	Xinmi +++	1	Saint Lucia +++	1	Saint Lucia +++
2	Linqing !!!	2	Gongyi +++	2	Grenada +++	2	Seychelles !!!
3	Gongyi +++	3	Yongcheng +++	3	Saint Kitts and Nevis !!!	3	São Tomé and Príncipe +++
	.....		.....		.....		.....
15	Linyi ---	15	Xinzheng +++	31	Cape Verde +++	31	Sierra Leone +++
16	Tongchuan ---	16	Xinxiang ---	32	Seychelles !!!	32	Guinea-Bissau +++
17	Linxia !!!	17	Xingyang +++	33	Barbados !!!	33	Liberia +++
18	Hanzhong ---	18	Changge +++	34	Trinidad and Tobago !!!	34	Comoros +++
19	Xingyang +++	19	Dengfeng +++	35	Cape Verde +++	35	Mauritius +++
	.....		.....		.....		.....

Positive Seeds	Negative Seeds	Positive Seeds	Negative Seeds
Xiangcheng	Shuozhou	Saint Vincent and the Grenadines	Paraguay
Linzhou	Zhangye	Guinea	Colombia
Yanshi	Lanzho	Libya	Bolivia
Weihui	Pu'er		
Mengzhou			
Positive Attributes		**Positive Attributes**	
<Province>	Henan	<Per_Capita_Income>	Low
Negative Attributes		**Negative Attributes**	
<Prefecture>	Perfecture-level city	<Continent>	South America

Figure 13. Case studies of GenExpan and the chain-of-thought reasoning technique. +++ and - - - stand for positive and negative target entities, respectively. !!! represents irrelevant entities belonging to the same fine-grained semantic class as the seed entities.