Assignment 2: Knowledge Graph Population

Mater Student Evangelia P. Panourgia, f3352402
Professor Dr. Panos Alexopoulos

General Comments

Regarding the LLM model we used **gpt-3.5-turbo** and for the Chat Completion Request, we used the following common setting :

```
response = self.client.chat.completions.create(
    model="gpt-3.5-turbo",
    temperature=0,
    messages=[
        {"role": "system", "content": self.system_message},
        {"role": "user", "content": user_message}
    ]
)
```

where :

- The code sends a request to the OpenAI API using the GPT-3.5-turbo model to generate a chat response.
- `response`: Stores the response received from the API call.
- `self.client.chat.completions.create`: The method that creates a chat completion using the API.
- `model="gpt-3.5-turbo"`: Specifies that the GPT-3.5-turbo model should be used.
- `temperature=0`: Sets the response generation to be deterministic, with no randomness.
- `messages=[]`: A list containing the conversation history (system and user messages).
- `"role": "system"`: Represents the system message setting up the assistant's behavior.
- `"content":` *self.system_message*: Provides the predefined system message content.
- `"role": "user"`: Represents the user's input message in the conversation.
- `"content": user_message`: Passes the user's message as input to the model.

In addition, the common **sysyem_message** is **"You are a relation extractor who can detect and classify relations between entities in text."**.
Furthermore ,in the prompts of this report are presented specific variable that is:

- **subj_entity**, **obj_entity**: represent the subject entity and object entity respectively for each of the rows of the given input file.
- {', '.join(self.**RELATIONS**)}: represents the pre-defines relation that is "cities_of_residence", "employee_of", "schools_attended", "spouse".
- {text}: represent the provided input sentence for which we want to extract the relation.

In addition, due to random existing in LLM-prompt process, during the reproduction, the results e.g. report for precision, recall, confusion matrix or error analysis samples may present different numbers.
Regarding the evaluation of the multiple LLM-promts for the purpose of the classification of the extracted relation, we used "precision" , "recall" and especially **"confusion matrix"** as we found the latter more easy regarding the explanation of our LLM model behaviour.

Last but not least, despite th fact that LLM is a "black-box" meaning that we don't know the internal decision process to label a sentence to a relation category, we created ,in our data frame, an additional column named "gpt_explanation" in case that it might express useful information regarding the "internal process" (which may be wrong ..). But, in practice, we observed that this column did not help us.

Replication Code/Dataset

For "replication" of this work, you can run (via selecting run all cells) the jupyter notebook named *KG&LLM_AUEB_Assignment_2.ipynb* (the results of multiple re-executions will be different to some extent due to randomness factor e.g. LLM classifiers don't return each time the same results). For the running process, we can use either "Google Colab" platform or locally "Jupyter Notebook", the advantage of the first one is the pre-installed python libraries. Otherwise, we should install the packages with the command "!pip install package_name". Regarding the generated dataset for advice / wish and uncertainty, the files are named *advice_wist.tsv* and *uncertainty.tsv* respectively. The integrated dataset derived from the aforementioned ones is named *advice_wish_uncertainty.tsv*.

Note 1: The Jupyter notebook contains in order both of the two assignments that is Task 1 and Task 2.

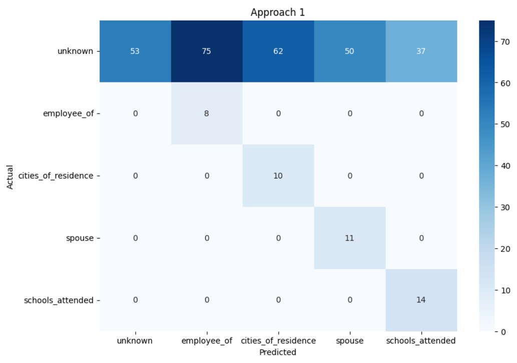
Note 2: The current Jupyter has written to read the given dataset named 'assignment3_dataset.tsv' from a folder 'data'.

Task 1

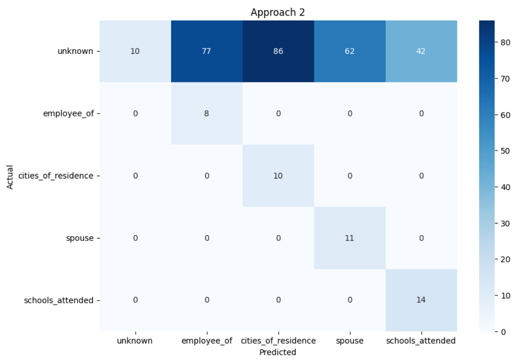
Before delving into the classification of predefined relations using the LLM classifier, we will first explain **Table: 1** and **Figure: 1**, as both are referenced multiple times in this section.

employee_of	cities_of_residence	schools_attended	spouse
1. Interview Ambiguity: This category includes instances where the information provided pertains to an interview, assessment, or employment inquiry but lacks confirmation of a person's employment status. **2. Possibility/Opinion Instead of Fact:** This category includes sentences that express opinions, assumptions, or speculation regarding a person's employment status, without confirming it as a fact. **3. Offer or Suggestion Without Employment Confirmation:** This category covers sentences mentioning job offers or suggestions without confirming whether the person accepted the offer or is currently employed.	**1. Future or Aspirational Statements:** Sentences expressing plans or dreams about living in a city rather than confirming actual residency. **2. Hypothetical or Conditional Statements:** Statements that describe hypothetical scenarios instead of actual events of living in a city. **3. Negative Statements:** Sentences explicitly stating that a person has never lived in a particular place.	**1. Motivation for Education** Sentences that convey a person's enthusiasm, eagerness, or determination regarding their educational aspirations. **2. Future Education Intent:** Sentences that express a person's plans or intentions to pursue education at a specific institution in the future.	**1. Future Event or Intention to Marry:** Describes future plans, intentions, or upcoming events where two people may become spouses **2. Non-Marital Relationship** Describes significant relationships between two people that are meaningful but do not establish them as spouses. **3. Explicit Negation of Marriage:** States that two people have not been married and are not currently married, contradicting any notion of them being spouses.

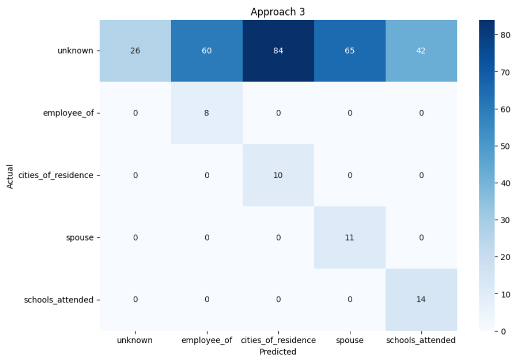
Table 1: Empirical Error Categories per Relation



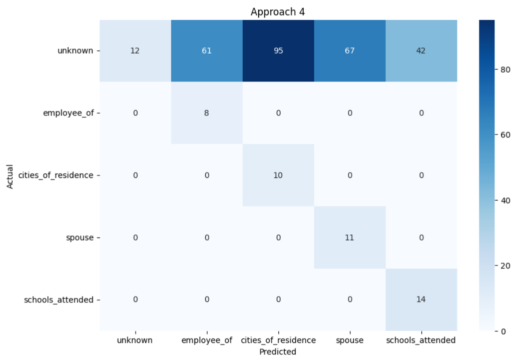
(a)



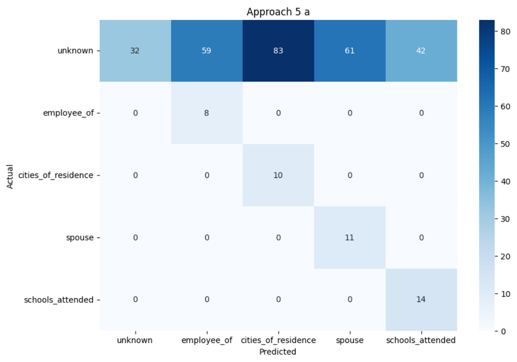
(b)



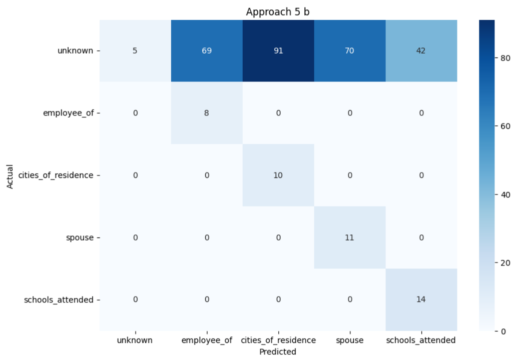
(c)



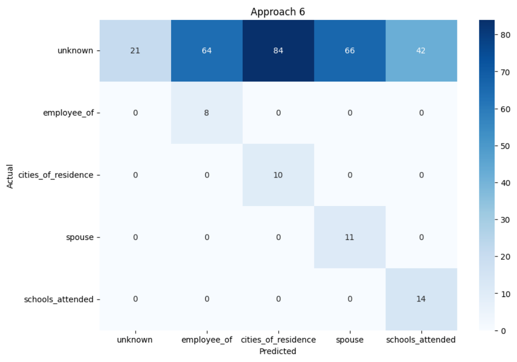
(d)



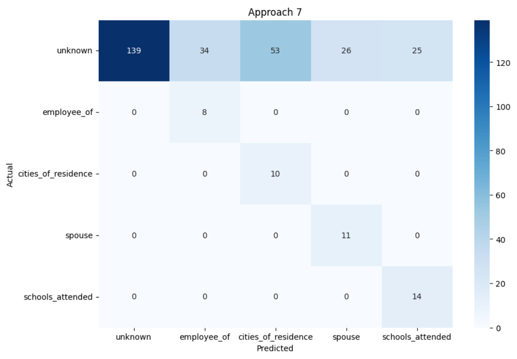
(e)



(f)



(g)



(h)

Figure 1: Confusion Matrices for all approaches prompts:**Zero-shot learning**: providing task **(a)** providing task and additional information per relation **(b)**. **Few-shot learning**: for **(c)** built on (b), with the addition of error examples (1 relation), for **(d)** built on (c), with the addition of defining error categories, for **(e),(f)** being similar to (c),(d), but 2 relations analysed, for **(g)** providing more examples for all error, for **(h)** built on (g) and focus on common errors (negative, future) and more examples.

The **Table 1.** presents the error categories identified through manual error analysis of random samples for each type of relation error. All of relation errors related to the class "Unknown", meaning for example that the provided text classified as "employee_of but the actual value had to be "Unknown". Each column displays the detected error categories corresponding to each relation type. For instance, in the first column, which represents the "employee_of" relation, three error categories are identified: "Interview Ambiguity," "Possibility/Opinion Instead of Fact," and "Offer or Suggestion Without Employment Confirmation." For each of these categories, an explanation or definition is provided. The remaining columns in the table follow the same format.

Regarding the **Figure: 1** illustrates the confusion matrices resulting from each approach applied to each adapted prompt (all approaches will be described in detail). We use confusion matrices as evaluation tools because they allow us to easily identify the categories that our LLM classifier misclassifies. In brief, a confusion matrix is a table used to assess the performance of a classification model by displaying the counts of true positives, true negatives, false positives, and false negatives. The diagonal elements indicate correctly predicted instances, while the off-diagonal elements represent errors made by the model. To interpret it effectively, one should compare the counts of true positives (correct predictions) and false positives/negatives (misclassifications) to evaluate the model's accuracy and identify error patterns.

By using only "recall" and "precision" metrics, it was challenging to pinpoint the specific tuples or pairs of classes where our LLM classifier made incorrect predictions.

For the scope of Task 1, we tested multiple prompts to enhance the classification of relations using the GPT-3.5-turbo LLM model. Initially, we applied **zero-shot learning** by running a basic prompt that outlined the specific task our LLM classifier was expected to perform. In the second prompt, we provided more information regarding the definition of each relation and we provided the constraint to based solely on the pre-defined definitions (of each relation). This is important because we wanted our classifier to categorize the data according to our specific definitions, rather than relying on any pre-existing or default classifications (in practice, however, it might not "understand" them). In figure 1, we can see the confusion matrices of all approaches. But, at this point, we will focus on (a), (b). From both confusion matrices, we can understand that our LLM classifier did not face problems of classification among the pre-defined relations that are: "employee_of", "cities_of_residence", "spouse", "schools_attended" as all values related to these "cells" are equal to zero (0). But, our classifier makes many mistakes in the following four (4) cases:

- The "employee_of" relation was incorrectly predicted as "unknown," indicating a misclassification.
- The "cities_of_residence" relation was incorrectly predicted as "unknown," indicating a misclassification.
- The "spouse" relation was incorrectly predicted as "unknown," indicating a misclassification.
- The "schools_attended" relation was incorrectly predicted as "unknown," indicating a misclassification.

Furthermore, in prompt 2, when we expressed the definitions of each relation, we observe that our LLM classifier made more mistakes. This may be due to the model's reliance on limited external knowledge from other datasets. However, we will not continue with zero-shot learning given the low performance scores.

- **Prompt 1**:

  Prompt for Approach 1 (Zero-shot learning)

  "In the following text, what is the relation that holds (or used to hold) between the entities *'subj_entity'* and *'obj_entity'*? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship."

- **Prompt 2** : Providing detailed information regarding the definition of our pre-defined relations.

Prompt for Approach 2 (Zero-shot learning)

"In the following text, what is the relation that holds (or used to hold) between the entities *'subj_entity'* and *'obj_entity'*? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship.
Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

If none of these relationships apply, respond with "Unknown". Otherwise, specify the relationship. "

After exploring zero-shot learning, we shifted our focus to **few-shot learning**, which requires providing examples in the prompt. To determine suitable examples, we conducted an error analysis to identify and focus on specific types of errors. We first investigated one of the categories most frequently misclassified by the model: the "employee_of" relation as we wanted to build our prompts step-by-step from analysing few information related to 1 relation) to more relations (that is all). We filtered our DataFrame to retrieve records where the LLM classified the text as "employee_of" while the actual relation was labeled as "unknown" (Figure: 2), resulting in a total of about 80 rows. Given time constraints, we randomly sampled ten (10) rows (Figure: 3) to review the sentences that were misclassified. From this sample, we identified and categorized the errors, which are presented in Table 1, specifically under the column for the "employee_of" relation.

```
filtered_employ_of = df_zero_shot_2[
    (df_zero_shot_2['gpt_relation'] == 'employee_of')
    &
    (df_zero_shot_2['relation_in_sentence'] == 'unknown')]
```

Figure 2: Filtering rows based on conditions in a DataFrame

```
    error_analysis_set_sample_filtered_employ_of = (
    'John was interviewed as part of the Apple company.',
    "It's possible that John is employed by Apple.",
    "John would be a perfect fit as an employee of Apple.",
    "John was not on Apple's payroll.",
    "John got a job offer by Apple",
    "John would excel as an employee of Apple.",
    "John has never been employed by Apple.",
    "Apple gave John an offer for a job.",
    "John has never been an Apple employee."
)
```

Figure 3: Sampled rows for the relation "employee_of"

Having identified potential sentences that may cause confusion, we expanded upon Approach 2 (that is prompt 2). Additionally, for errors associated with the categories "unknown" and "employee_of," we provided six examples for each category. Examining the confusion matrices (b) and (c) in Figure 1, we can see that our model has **improved**, correctly predicting more instances as "unknown" (increasing from 10 to 26) and reducing incorrect predictions of "employee_of" (decreasing from 77 to 60). Regarding the remaining misclassifications, there are no significant changes. Furthermore, to verify that our modeled indeed "learned" from our examples, we check how many instances from the sample predicted correctly, as reminder, at the beginning all predictions of the sample were predicted wrong. In Figure 4, we can see that predicted 2 instances correctly (rows: 4,5), so it appeared a slight improvement in our sample data.

```
   df_few_shot_3[["relation_in_sentence","gpt_relation"]][df_few_shot_3['text'].isin(
   error_analysis_set_sample_filtered_employ_of)]

    Output:
    relation_in_sentence   gpt_relation
        unknown                employee_of
        unknown                employee_of
        unknown                employee_of
        unknown                unknown
        unknown                unknown
        unknown                employee_of
        unknown                cities_of_residence
        unknown                employee_of
        unknown                employee_of
```

Figure 4: Test based on sample for teh relation "employee_of"

- **Promt 3:**

Prompt for Approach 3 (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities $'subj\_entity'$ and $'obj\_entity'$? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

If none of these relationships apply, respond with "Unknown". Otherwise, specify the relationship.

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify text data as "employee_of". but, the actually label is "unknown".

So, we provide six examples that should b classifieds as "unknown" instead of "employee_of":

1. Evangelia completed an interview with Amazon
2. George answered the technical interview questions as part of the Apple company.
3. Sarah would thrive as a team leader at Google.
4. George has the skills to succeed as an employee at Google.
5. Sarah received a job offer from Microsoft.
6. Emily was extended an offer to join Amazon.

Please, classify the given text to the pre-defined categories. "

For additional "potential" improvement, we wrote a new prompt combining both the definitions of error categories based on Table: 1 (see: column "employee_of") and we provided an example for each (representing in prompt 4). At this point, to remind, that our general aim was to express our "prompt" with the "best" way (given our time limitation of our assignment, as finding a good prompt requires many trial-errors, and as a result it might be a time-consuming process). The "best" prompt is a relative term meaning the best scores (recall, precision, confusion matrix) given our time constraints of the assignment.

Prompt for Approach 4 (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities $'subj\_entity'$ and $'obj\_entity'$? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

If none of these relationships apply, respond with "Unknown". Otherwise, specify the relationship.

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify text data as "employee_of". but, the actually label is "unknown".

So, we provide the error categories and examples for each.
1. Interview Ambiguity:

Definition: This category includes instances where the information provided pertains to an interview, assessment, or employment inquiry but lacks confirmation of a person's employment status. Phrases may mention "participation," "interviews," or "employment inquiries" without providing evidence of an employment relationship.
Example: "Evangelia completed an interview with Amazon."
2. Possibility/Opinion Instead of Fact:

Definition: This category includes sentences that express opinions, assumptions, or speculation regarding a person's employment status, without confirming it as a fact. These statements may suggest a possibility, express a subjective viewpoint, or predict someone's suitability for a role but do not provide factual employment details.
Example: "Sarah would thrive as a team leader at Google."
3. Offer or Suggestion Without Employment Confirmation:

Definition: This category covers sentences mentioning job offers or suggestions without confirming whether the person accepted the offer or is currently employed. It implies potential future employment but does not establish an existing employment relationship.
Example: "Sarah received a job offer from Microsoft."

Task:
Please, classify the given text to the pre-defined categories.

Comparing the confusion matrices of (c), (d) in 1, we observed that the LLM classifier appeared discouraged results in across all classes, as the the number of correct predicted instances as "unknown" decreased, whereas the incorrect predictions for the remaining classes increased. However, to be sure that the combination of providing both definitions for error categories and examples for each, it "surely" doesn't work, we will extend it for **two relations** that is "employee_of" (the already analysed) and "cites_of_residence" as the most misclassification are presented in these ones. For this investigation, we wrote the prompts 5.a. (containing only examples for each error category) and 5.b. (containing both definition for error categories and examples for each). At this point, we have to mention that we took a random sample for the relation "cites_of_residence" in order to extract the error categories (similarly to "employee_of"). Note, the sample pulled from the data frame related to (c), as we built on it (of course we could take the sample in our initial data frame and as a result it would be another strategy trial-and-error method, but due to time limitations we only followed the sample process as it described).

Regarding the evaluation process, comparing the confusion matrices (e), (f) in 1, we can observe that (e) ,that is the prompt related to prompt 5.a., has better performance as it predicts correctly more "unknown" classes (32 whereas (f) only (5)!), and regarding the majority of the remaining classes the incorrect predictions decreased (e.g. in (e) 59 for "employee_of", whereas in (f) 69 for "employee_of", meaning that in (f) LLM classifies more sentences as "employee_of" incorrectly).

Therefore, we decided to apply the method of providing only specific examples for each incorrectly classified relation across all types of relations.. In other words, we empirically understood that the additional information related to the description LLM classifier caused problems to its performance, so we removed it (we removed the "noise").

Prompt for Approach 5.a. (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities $'subj\_entity'$ and $'obj\_entity'$? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify incorrectly text data as "employee_of" or as "cities_of_residence". But the correct answer was "unkown". So, we provide examples of sentences that you made mistakes.

Misclassification errors for the type of errors related to the relation "employee_of"
Examples: "Evangelia completed an interview with Amazon.","Jack solved the whole code task during the Interview for Microsoft", "Sarah would thrive as a team leader at Google.", "Sarah received a job offer from Microsoft."

Misclassification errors for the type of errors related to the relation "cities_of_residence"
Examples: "Maria dreams of living in Paris one day.", "If Alex had moved to Tokyo, he might have pursued a different career.", "Samantha has never lived in New York City.", "Evangelia did not visited Athens."

Task:
Please, classify the given text to the pre-defined categories.

Prompt for Approach 5.b. (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities $'subj\_entity'$ and $'obj\_entity'$? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify text data as "employee_of" or incorrectly classified as "cities_of_residence". So, we provide the error categories and examples for each.

Misclassification errors for "employee_of"
1. Interview Ambiguity:

Definition: This category includes instances where the information provided pertains to an interview, assessment, or employment inquiry but lacks confirmation of a person's employment status. Phrases may mention "participation," "interviews," or "employment inquiries" without providing evidence of an employment relationship. Example: "Evangelia completed an interview with Amazon."
2. Possibility/Opinion Instead of Fact:

Definition: This category includes sentences that express opinions, assumptions, or speculation regarding a person's employment status, without confirming it as a fact. These statements may suggest a possibility, express a subjective viewpoint, or predict someone's suitability for a role but do not provide factual employment details.
Example: "Sarah would thrive as a team leader at Google."

3. Offer or Suggestion Without Employment Confirmation:

Definition: This category covers sentences mentioning job offers or suggestions without confirming whether the person accepted the offer or is currently employed. It implies potential future employment but does not establish an existing employment relationship.
Example: "Sarah received a job offer from Microsoft."

Misclassification error for "cities_of_residence"

1. Future or Aspirational Statements:

Definition: Sentences expressing plans or dreams about living in a city rather than confirming actual residency. Example: "Maria dreams of living in Paris one day."

2. Hypothetical or Conditional Statements:

Definition: Statements that describe hypothetical scenarios instead of actual events of living in a city.
Example: "If Alex had moved to Tokyo, he might have pursued a different career."

3. Negative Statements:

Definition: Sentences explicitly stating that a person has never lived in a particular place, indicating that it should not be categorized as "cities_of_residence."
Example: "Samantha has never lived in New York City."

Task:

Please, classify the given text to the pre-defined categories.

We extended the prompt to cover all relations, as up to this point, we had only focused on "employee_of" and "cities_of_residence." To do this, we first conducted an error analysis by taking random samples of 10 instances for each of the remaining relations and manually classifying the errors (similar process ti the samples related to the the already examined relations). This enabled us to expand our predefined strategy (see bold letters in the page 9.) to include the remaining relations ("school_attended" and "spouse"). In other words, we provided error examples for all misclassified relations. The adapted prompt (Prompt 6) is shown in the figure below.

Prompt for Approach 6 (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities *'subj_entity'* and *'obj_entity'*? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify incorrectly text data as "employee_of" or as "cities_of_residence" or as "spouse" or as "schools_attended". But the correct answer was "unknown". So, we provide examples of sentences that you made mistakes.

Misclassification errors for the type of errors related to the relation "employee_of"
Examples: "Evangelia completed an interview with Amazon.","Jack solved the whole code task during the Interview for Microsoft", "Sarah would thrive as a team leader at Google.", "Sarah received a job offer from Microsoft."

Misclassification errors for the type of errors related to the relation "cities_of_residence"
Examples: "Maria dreams of living in Paris one day.", "If Alex had moved to Tokyo, he might have pursued a different career.", "Samantha has never lived in New York City.", "Evangelia did not visited Athens."

Misclassification errors for the type of erros related to the relation "spouse"
Examples: "I may get married to Leo in near future.", "Jane and George has a child.", "Frida has a child with George", "I am not married to George.", "Maria is not wife of Mike."

Misclassification errors for the type of errors related to the relation "schools_attended"
Examples: "I am eager to study computer science in MIT.", "Leo is willing to study in Imperial", "I wil study marketing in MIT in a few months.", "Sara may be master student in Columbia."

Task:
Please, classify the given text to the pre-defined categories.

Comparing the confusion matrices (e) and **(g)** in Figure 1, it is evident that our LLM classifier shows an increase in incorrect predictions with the new approach. Specifically, the number of correctly predicted "unknown" relations decreased from 32 in matrix (e) to 21 in matrix (g), and incorrect classifications increased for other categories (e.g., for the "employee_of" relation, errors rose from 59 in (e) to 64 in (g)). Although these differences are not quite substantial, we plan to further analyze our data to refine our prompts based on the existing strategy. Note, at this point we have manually analysed samples for error categories for **all** relations.

Having at our disposal the Table 1., we tried to leverage it with purpose to extract "potential" patters of categories of errors **across all** relations. We observed that across columns there were some commonly error categories related to **"future"** and **"negative" meanings**. Before adapting our prompt with these "insights", we analysed our data more. Firstly, we filtered our dataset in the column "text" with regex expressions related to "negative" (Figure: 5) and "future" (Figure: 6) meanings, because we wanted to verify that the actual value of these sentences / text should be "unknown". Indeed, we verified that the actual label in the **whole** dataset for these meanings was **unknown**. Furthermore, applying in depth analysis in our given dataset related to our last approach (that is prompt 6), we filtered it with purpose to analyse only the lines related to "future meaning", and we indeed observed that all of them had classified wrongly (meaning with one of the pre-defined relations, whereas the actual relation was "unknown"), the related code is in Figure 7. Similarly, we repeated the process for the text related to "negative" meaning, the related code is in Figure 8, and our insight was similar to the aforementioned one. Both of them consist strong evidence that our LLM should be improved regarding the sentences containing either "future" or "negative" meanings for the given prompt. So, our empirical "pattern" stemming from sample error categories is valid for the "whole" given dataset. For additional information, the "future" meaning represents at least 8% of our whole dataset (we say, at least as we may lose some future sentences in case that they expressed with an uncommon way e.g. If they not use "will" and as a result the regex expression cannot "catch" it). Regarding "negative" meanings, we calculated a higher coverage percentage compared to the aforementioned one , being at least 25%, and 20% of them has classified incorrect as one of the pre-defined relations instead of the actual valued being "unknown". So, based on this basic data analysis, if we succeed in improving our LLM prompt, we will wait a high improvement as these cases together cover about (1/3) of our whole dataset.

So, we extended the already prompt (prompt 6), via passing error examples for all relations and additional via expressing the common mistake related to future and negative relations, for the latter we guided the LLM to assign the value "unknown". In addition, we tried to "give" time to our LLM model to "think" via separating in clear steps the task ("Firstly", "Otherwise"). In the following box, the adapted prompt is presented (Prompt 7).

```
    # Define a regex pattern for negative expressions
    pattern = r"\b(?:no|not|never|none|nobody|nothing|neither|nowhere|can't|
    cannot|won't|wouldn't|shouldn't|couldn't|don't|doesn't|didn't|isn't|a
    ren't|wasn't|weren't|hasn't|haven't|hadn't|mustn't)\b"

    # Filter rows in the DataFrame that contain negative expressions in the 'text' column
    negative_sentences = df[df['text'].str.contains(pattern, flags=re.IGNORECASE, regex=True)]
    negative_sentences
```

Figure 5: Filtering and regex expressions for "negative meaning"

```
    # Define a regex pattern for future negative sentences
    # It looks for patterns indicating future tense (will, going to, etc.) combined
    #with negative words (not, won't, etc.)
    pattern = r'\b(?:will|going to)\b\s*'

    # Apply the pattern to filter the rows with future negative sentences
    future_negative_sentences = df[df['text'].str.contains(pattern, flags=re.IGNORECASE, regex=True)]
```

Figure 6: Filtering and regex expressions for "future meaning"

```
   set_for_future_sentences = set(list(future_sentences.text))
   # focus on the columns "relation i nsentence" and "gpt relation" are diffrent!
    df_few_shot_6[df_few_shot_6['text'].isin(set_for_future_sentences)]
    (future_sentences["relation_in_sentence"].value_counts().values[0]/df.shape[0])*100
    # Output: 7.8125
```

Figure 7: Data Analysis for "future meaning"

```
   set_for_negative_sentences = set(list(negative_sentences.text))
   check_correct_predictions_for_negative = df_few_shot_6[df_few_shot_6['text'].isin(
   set_for_negative_sentences)]
    check_correct_predictions_for_negative[check_correct_predictions_for_negative[
    'relation_in_sentence']!= check_correct_predictions_for_negative["gpt_relation"]]
    print(f"'The percenatge of incorrect classified sentences related to negative meaning',{
    (incorrect_predictions_negative_sentences.shape[0]/df.shape[0])*100
    }'%'")
    # Output:
    'The percenatge of incorrect classified sentences related to negative meaning',21.25'%'
```

Figure 8: Data Analysis for "negative meaning"

Before analysing the evaluation scores in depth meaning "precision", "recall" and "confusion matrix" (as this seems to be our "best" approach so far), we firstly checked, if our LLM classifier model "learned" the prompt related to "future" and "negative" meanings. For this purpose, we filtered again with the regex expression (expressing both future and negative meanings) in the new data frame (that is df_few_shot_7) related to the predictions stemming from the prompt 7, and we observed that regarding the "future" meanings, the predictions are all correct, whereas regarding the "negative" meanings, 16% of them were predicted incorrectly (being a low incorrect percentage of predictions). So, we can say that our LLM "learned" to a high degree our prompt.

Comparing the confusion matrices (g) and **(h)** in Figure 1, we observed a significant improvement in the performance of our LLM classifier. This is obvious from the pallet color of blue, meaning that in (h) the correctly predictions for the "unknown" relation are depicted with bold blue (from 21 increased to **139** correct predictions). Also, In (h), the incorrect predictions for the pre-defined relations, all decreased. More specifically:

- The number of incorrect predictions for the "employee_of" relation decreased from 64 in (g) to 34 in (h).
- The number of incorrect predictions for the "cities_of_residence" relation decreased from 84 in (g) to 53 in (h).
- The number of incorrect predictions for the "spouse" relation decreased from 66 in (g) to 26 in (h).
- The number of incorrect predictions for the "schools_attended" relation decreased from 42 in (g) to 25 in (h).

Furthermore, regarding "precision", "recall" and "F1 score" our "best" prompt (due to time constraints we cannot investigate more our data), based on the following Table (Table 1.), we can say the following ones for our LLM classifier: The model performs well in terms of recall across the pre-defined relations (except "unknown" relation), successfully identifying all true instances. However, precision is notably low (except "unknown" relation), especially for relations like "employee_of" and "cities_of_residence." This imbalance results in moderate F1 scores overall, suggesting that while the model captures all relevant instances, it often incorrectly classifies other instances as belonging to these relations, which has been shown in our confusion matrix, too. Improving precision is essential for enhancing the overall performance of the model across these categories.

Table 1: Relation Metrics for The Best Prompt

Relation	**Precision**	**Recall**	**F1 Score**
unknown	1.000000	0.501805	0.668269
employee_of	0.190476	1.000000	0.320000
cities_of_residence	0.158730	1.000000	0.273973
spouse	0.297297	1.000000	0.458333
schools_attended	0.358974	1.000000	0.528302

Prompt for Approach 7 (Few-shot learning)

In the following text, what is the relation that holds (or used to hold) between the entities $'subj\_entity'$ and $'obj\_entity'$? The relations can only be one of the following: {', '.join(self.RELATIONS)} Text: {text} Please, if none of these relationships apply, respond with "Unknown" otherwise, specify the relationship. Furthermore, consider solely the following definitions for the pre-defined relation:

- "cities_of_residence": relates a person to the cities they currently live or have lived in the past
- "employee_of": relates a person to the organizations they are currently employees of or have been in the past
- "schools_attended": relates a person to the schools they are currently attending or have attended in the past.
- "spouse": relates a person to the persons they are currently married to or have been married to in the past

In addition, We observed that you made mistakes in the category "unknown" as you tend to classify incorrectly text data as "employee_of" or as "cities_of_residence" or as "spouse" or as "schools_attended". But the correct answer was "unknown".
So, we provide examples of sentences that you made mistakes. The correct answer should be for all of the following examples "unknown".

Examples of incorrect classification:
1."Evangelia completed an interview with Amazon.",
2. "Jach solved the whole code task for Amazon",
3."Sarah would thrive as a team leader at Google.",
4. "Sarah received a job offer from Microsoft",
5."Maria dreams of living in Paris one day.",
6."If Alex had moved to Tokyo, he might have pursued a different career.",
7."Samantha has never lived in New York City.",
8."Evangelia did not visited Athens."
9."I may get married to Leo in near future.",
10."Jane and George has a child.",
11."Frida has a child with George",
12."I am not married to George.",
13."Maria is not wife of Mike."

In addition, we observed that you made mistakes in sentences containing :
1. "future" meaning expressing with the following words: "will","possible","are going to","is going to", "soon" or
2. negative meaning expressing with the following words:
"no","not","never","none","nobody","nothing","neither","nowhere",
"can't","cannot","won't","wouldn't","shouldn't","couldn't","don't","doesn't","didn't",
"isn't","aren't","wasn't","weren't","hasn't","haven't","hadn't","mustn't".

So, Classify negative and future sentences as "unknown".
Task:
Please, Firstly, Check if the text containing future or negative words, if yes, then classify the text as "unknown".
Otherwise, classify the text based the pre-defined definitions. If none of the pre-defined relationships apply, respond with "unknown".

Task 2

To construct the datasets related to uncertainty and advice/wish, we utilized ChatGPT, with the dialogue interactions hosted on [1]. The code for generating these datasets is available in the Jupyter notebook under the section titled "Task 2 (Generated Dataset)." The core logic involves creating a predefined template structured as a dictionary, accompanied by predefined lists containing values for names of people, cities, organizations, schools, and spouses. The output of this code consists of two tsv files: uncertainty.tsv and advice_wish.tsv, containing 48 and 51 rows of data, respectively.

The datasets were initially separated for debugging (and evaluation) purposes. Although all generated sentences fall under the category "unknown," each dataset was separated based on a specific relation (e.g., "employee_of") to trace which relation each sentence was derived from. However, in the final integrated dataset, the relation was correctly updated to "unknown." This integration process was done manually.

The two datasets were subsequently combined into a final file named advice_wish_uncertainty.tsv, which contains a total of 99 generated data entries. To ensure the uniqueness of our dataset compared to the given dataset (assignment3_dataset.tsv), we first saved the values from the "text" column of our initial data frame into a set data structure. We then checked whether these values were present in our newly generated data frame to verify their uniqueness 9. Additionally, we performed a duplicate check on our new data frame using the duplicated function to confirm that it does not contain any repeated values.

```
    check_values = set(list(df["text"]))
    boolean_uniqness_advise_wish = list(df_few_shot_8_advice_wish['text'].isin(check_values))
    [i for i in  boolean_uniqness_advise_wish if i == True]
    boolean_uniqness_uncertainty = list(df_few_shot_8_advice_wish['text'].isin(check_values))
    [i for i in  boolean_uniqness_uncertainty if i == True]
```

Figure 9: Check uniqness of our Dataset

For the evaluation of our LLM classifier on the newly generated datasets, we conducted separate runs for each subset: one containing the generated data for uncertainty and the other for advice/wish. An additional preprocessing step was applied to accurately define the actual relation labeled as "unknown." We also ran the evaluation on the combined dataset, which integrates both subsets. This approach allowed us to assess whether either of the individual subsets had a greater impact on the overall performance of our LLM classifier.

The Tables 2, 3 refer to the sub data set related to advice / wish. Firstly, analysing the values of the confusion matrix, we see that 16 values correctly predicted as "unknown", whereas 35 values in total predicted incorrectly. A general comment, the reason why we have data only in the first row related to "unknown" is because our generated data set for advice / wish contains only relation with actual value "unknown". For the values of precision / recall table, we can observe that for the relation "unknown" the precision is 1, this means that all instances predicted as "unknown" by the model were actually correct. A precision of 1.0 indicates perfect precision; there were no false positives for this relation. But, the recall is equally to 0.30 indicating that the model only captured around 30% of all the actual "unknown" instances in the dataset. This suggests that while the model is highly accurate when it predicts "unknown," it fails to identify a significant portion of true instances of this relation, leading to a large number of false negatives.

Table 2: Confusion matrix for the sub data set Advice / Wish

Relation	**unknown**	**employee_of**	**cities_of_residence**	**spouse**	**schools_attended**
unknown	16	9	8	7	11
employee_of	0	0	0	0	0
cities_of_residence	0	0	0	0	0
spouse	0	0	0	0	0
schools_attended	0	0	0	0	0

Table 3: Precision and Recall for the sub data set Advice / Wish

Relation	**Precision**	**Recall**
unknown	1.0	0.313725
employee_of	0.0	0.000000
cities_of_residence	0.0	0.000000
spouse	0.0	0.000000
schools_attended	0.0	0.000000

The Tables 4, 5 refer to the sub data set related to Uncertainty. Firstly, analysing the values of confusion matrix, we see that 22 values correctly predicted as "unknown", whereas 26 values it total predicted incorrectly. This indicates that our LLM classifier predict correctly approximately the half number of sentences. However, in this case our classifier seems to appear a bit better performance observing the recall value.

Generally, we observe that our LLM classifier make many mistakes in both datasets.

Table 4: Confusion matrix for the sub data set Uncertainty

Relation	**unknown**	**employee_of**	**cities_of_residence**	**spouse**	**schools_attended**
unknown	22	9	10	1	6
employee_of	0	0	0	0	0
cities_of_residence	0	0	0	0	0
spouse	0	0	0	0	0
schools_attended	0	0	0	0	0

Table 5: Precision and Recall for the sub data set Uncertainty

Relation	**Precision**	**Recall**
unknown	1.0	0.458333
employee_of	0.0	0.000000
cities_of_residence	0.0	0.000000
spouse	0.0	0.000000
schools_attended	0.0	0.000000

After analyzing the datasets individually, we can now observe the results of integrating data from both synthetic data sets in Tables 6, 7. For the "unknown" relation in confusion matrix, the model correctly identifies 44 instances as "unknown", but it misclassifies instances of unknown as other relations: 16 as employee_of, 19 as cities_of_residence, 3 as spouse, and 17 as schools_attended. Regarding precision and recall, the model is perfect in predicting "unknown" when it does, meaning all predictions labeled as unknown were actually correct. But, observing the recall of the relation "unknown", the model only captures around 44.4% of the actual "unknown" instances, indicating that over half of the unknown instances are misclassified as other relations. So, our LLM classifier with our "best" prompt, so far, seems **not** to classify correctly a great number of sentences related to wish / advice and uncertainty.

Table 6: Confusion matrix for The Best Prompt

Relation	**unknown**	**employee_of**	**cities_of_residence**	**spouse**	**schools_attended**
unknown	44	16	19	3	17
employee_of	0	0	0	0	0
cities_of_residence	0	0	0	0	0
spouse	0	0	0	0	0
schools_attended	0	0	0	0	0

Table 7: Precision and Recall for The Best Prompt

Relation	**Precision**	**Recall**
unknown	1.0	0.444444
employee_of	0.0	0.000000
cities_of_residence	0.0	0.000000
spouse	0.0	0.000000
schools_attended	0.0	0.000000

Future Work

We could run more prompts based on the extracted "pattern" that is error related to "future" and "negative" (for example more future and negative examples), and via insisting on passing more examples for all categories of errors.

References

[1] ChatGPT Dialogue for generation of Dataset. Available at: `https://chatgpt.com/share/67162951-ae28-800e-aa8e-716c7d4848da`. Accessed: October 19, 2024.