Entity Linking to Dynamically-Evolving Personal Knowledge Graphs in Conversations

Abiram Mohanaraj
Aalborg University
Aalborg, Denmark
amohan17@student.aau.dk

Elisabeth Niemeyer Laursen
Aalborg University
Aalborg, Denmark
elaurs17@student.aau.dk

SUMMARY

In our last semester's work, we investigated conversational agents and their limitations to provide personalised responses to the user, i.e., responses that consider the user's information. To accomplish this, we observe a need for a structured representation of the personal information to enable the conversational agents to personalise responses. To this end, we explored an architecture of existing solutions to populate personal knowledge graphs (PKGs) by extracting relevant personal information from a conversation. PKGs are user-centered knowledge graphs (KGs), which model the user's information. These graphs are unique in that they only include information about a single user.

Our architecture from the previous semester's work consists of two phases, where the first is textual triple extraction, i.e., the extraction of two substrings or entity mentions from the input utterance and their relation. Second, we link the previously extracted entity mentions to an open KG. We observe a number of inadequacies when using this architecture, which is (1) the architecture does not effectively disambiguate references to personal entities from general entities. (2) The actual relation between the linked entity and the entity mention in the triple is not defined. For this semester, we aimed to improve upon this architecture by adding components responsible for linking entities to a PKG and resolving the triples to fit the PKG. Looking at the inadequacies in the existing architecture, we define two problems, which we refer to as PKG Statement Linking and PKG Enrichment. We concentrated on the PKG Statement Linking challenge, which entails linking each entity mention from the textual triples to PKG entities in a local PKG.

To solve the PKG statement linking problem, we require a new, specialised dataset, since no existing dataset contains conversations with textual triples, entity linking to open KG, and personal entity annotations. From our previous semester's architecture, we know that we require textual triple annotations of conversations, and entity linking annotation to an open KG. Additionally, specific to our problem, we also need annotations of personal entities, i.e., annotations of references to the same entity throughout a conversation or references to entities in a local PKG, constructable from the conversation. As the open KG for our dataset, we decided that a Common Sense Knowledge Graph (CSKG) would be more appropriate, as these capture basic knowledge like "a person has a head and two arms", which could benefit a conversational agent. We surveyed a number of different CSKGs and determine that *ConceptNet* is ideal for our use case. As a result, we have constructed a dataset for training and evaluating PKG population architectures with reliable ground truth data. To collect this dataset, we created three different websites to annotate the data. Each website corresponds to one of the annotations required for PKG population. These websites also makes it possible to expand the dataset for future research.

For the PKG Statement Linking problem, we present a pipelined architecture for this consisting primarily of two components, namely the Personal Entity Classifier (PEC) and the Personal Entity Disambiguator (PED). The goal of our PEC is to determine the presence of an entity mention in the PKG, while PED links the entity mention to the specific entity in the PKG. PEC is a modified transformer architecture, where we change the input embedding layer, and the encoder layer. The input embedding layer uses a super graph consisting of the PKG and Utterance Relation Graph (URG), which combines the textual triples with the utterance. This is a new data structure, we propose. We modify the encoder layer of the transformer to mask the input with the goal of enforcing the graph structure in the URG. On top of the transformer, we add a fully connected layer, which will output whether one particular entity mention in the textual triples is present in the input PKG. Dependent on this classification, we proceed to the PED component, which either creates a new entity or maps to an existing entity. If the entity is deemed to exist by the PEC, we map it using gestalt pattern matching. We differentiate between pronouns and other textual mentions, since linking pronouns are modelled as the coreference resolution problem.

We evaluate our architecture by comparing PEC to baseline models, where it outperforms them $35 - 42\%$ in F1, while we achieve a F1 of 0.87 in the PED experiment. We find that even as the solution performs well, it is not sufficient in a practical setting. Though we leave room for optimisations to our model, we have still managed to improve on the PKG population pipeline. This strongly encourages further research into the usage of KGs to structure personal information for conversational agents. We also offer explicit proposals for future research topics within this field, such as alternative approaches to personal entity linking and utilisation of PKGs.

ABSTRACT

A key challenge with conversational agents is providing user-tailored responses to the user. This includes structuring personal information about a particular user and referring to it at the right time. In recent research, we observe the usage of knowledge graphs to retrieve and store personal information as an ideal representation. From the previous semester work in populating user-specific knowledge graphs, we observe deficiencies in an architecture of existing solutions. The first deficiency is a lack of entity linking to local personal knowledge graph (PKG), which we refer to PKG Statement Linking problem. The second problem concerns updating the PKG with the information from the previous problem. We refer to this as PKG Enrichment. As both of these problem are complex, we focus on the former.

We also construct a dataset suitable for training and testing PKG population architectures, since we find no conversational dataset with annotations of textual triple, entity linking to open KG, and personal entities. The dataset spans 100 conversations with triple annotations, personal entity annotations, and ConceptNet entity annotations. We choose ConceptNet as the ideal common sense knowledge graph through a survey on the open knowledge graphs. To further collect the annotations, we make three annotation website implementations available.

Our proposed solution consists of a personal entity classifier (PEC) and personal entity disambiguator (PED). PEC determines whether non-pronoun entity mention in an utterance is present in the PKG. We propose a new transformer architecture with a modified input embedding and masking layer for this purpose. As input, we propose a supergraph consisting of a PKG and an Utterance Relation Graph (URG), which combines an utterance and textual triples (i.e., two substrings and the relation inbetween). We evaluate the PEC in comparison to two baseline models, where it outperforms them by $35-42\%$ in F1-score.

The second component, PED, is a heuristic-based method for linking the entity mention to a specific personal entity, which also incorporates coreference resolution for linking pronouns. In our experiment with PED, we achieve an F1 of 0.87. We observe that the critical component of the architecture is the PEC, as we observe a significant effect of compounding error in the performance of PED.

Though our findings display room for improvement, we have still managed to uncover vital information about the basis of personal knowledge graph population, which brings us one step closer to personalised conversation.

1 INTRODUCTION

Systems that can intelligently converse with humans, Conversational Agents (CAs), have become popular [1] with meaningful applications in personal assistants, information retrieval, and e-commerce to name a few [16, 27, 44, 46]. The

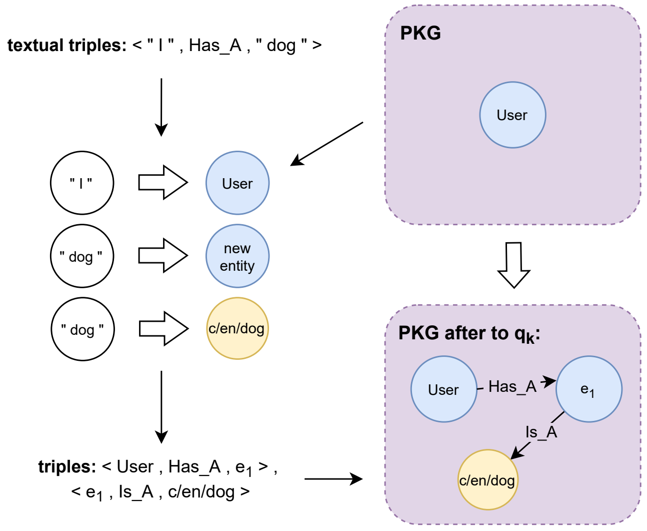

textual triples: < " I " , Has_A , " dog " >
PKG
User
" I "
User
" dog "
new entity
" dog "
c/en/dog
PKG after to q_k:
User
Has_A
e_1
Is_A
c/en/dog
triples: < User , Has_A , e_1 > , < e_1 , Is_A , c/en/dog >

Figure 1. Example of entity linking to PKG

CAs struggle to personalise responses to users, i.e., provide a meaningful response to a user, which is relevant to the conversation while also taking into account user-specific preferences or information [3, 8]. A plausible reason for the CAs lack of ability in providing personalised responses is that the models used in the CAs do not effectively capture the user-specific information from the conversations before responding to the user as the personal information tends to be implicitly represented [3, 8]. An approach to mitigate this reason would be to explicitly represent the personal information and then use this representation to provide personalised responses. One data representation is a Knowledge Graph (KG). KGs can be used as a method of storing information for CAs since they are easy to traverse due to their graph structure [52]. They are a collection of facts consisting of entities and relations, and each fact can be expressed as a factual triple of subject, predicate, and object [20]. They are also a machine-readable representation and enable CAs to use heterogeneous information [20]. This is important for the CA as it would enable the system to perform logical reasoning over the KG to personalise the responses. E.g., when a user says she likes a particular artist and we have a knowledge graph with information about the artist including collaborators, events, and so on, using this information in a response to the user would result in a personalised response.

KGs have also been utilised in similar systems to CAs, e.g., question answering [9, 17, 30]. However, these systems utilise an open KG on factual information. To personalise responses, we need a user-specific KG, constructed on conversations with a user. We regard this user-centric KG as Personal Knowledge Graph (PKG), which has the additional requirement that there must exist at least one path for each entity in the PKG containing both the entity and a special user entity [7]. PKGs have also been utilised as a representation to model personal information about particular users, where textual triples are extracted [7, 24]. The textual triples contain surface-form words as subject and object, i.e., phrases from the conversation. This means that entities can be represented multiple times. To this end, we need entity linking to map the phrases to the entities. Few works also explore entity linking in conversations [10, 21].

In our previous semester work, we examined population of PKGs and proposed an architecture of existing solutions for populating PKGs from conversations. We present this architecture in **Figure 2**, which centres on the two tasks, namely *Triple Extraction* and *Entity Linking*. We found this architecture to be successful in extracting personal information from conversational data using state-of-the-art triple extractor and entity linker.

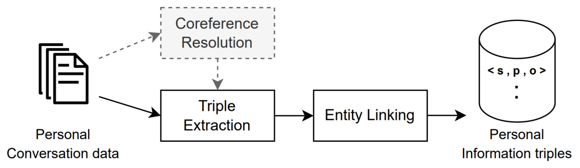


Figure 2. The architecture of existing solutions for populating PKGs with conversational data.

For **Figure 2**, triple extraction refers to the process of classifying and extracting personal information in the format of a textual triple, containing two entity mentions and the relation between them [17, 24, 49]. In the second phase, entity linking concerns linking the entity mentions to an open knowledge graph to infer more information about the entities[2, 21].

We observe a number of deficiencies when using this architecture, which is (1) the architecture does not effectively disambiguate between open KG entities and personal entities, i.e., an entity specific to a user. For example, for the information that the user has a dog named 'Fido' and thinks that big dogs, in general, are loud, this information is modelled as the user has a dog, which is big and loud using the architecture, which is not necessarily true. (2) The actual relation between the linked open KG entity and the entity mention in the triple is not properly defined. If the entity mentions are linked to a local PKG entity and the entity mention is also linkable to an open KG entity, we also need to determine how the PKG entity is related to the open KG entity. An example of this problems is illustrated in Figure 2.

In this work, we create a dataset with reliable ground truth annotations for training and testing PKG population architectures. We also improve upon the aforementioned architecture by adding components responsible for linking entities to a local PKG and resolving the triples to fit the PKG (i.e., the aforementioned deficiencies). We propose the following research questions:

1. How can we identify and link personal entities into the PKG?
2. How can we enrich the PKG with triples through entity linking to both the PKG and an open KG?

2 RELATED WORKS

Entity Linking. Identifying entity mentions and linking them to KG entities is formally defined as the problem of entity linking [11, 21, 39]. A popular line of thought for this task is to map the entity mentions to unique identifiers representing the KG entities [10, 28, 39, 51]. This is also a challenging task, where the current state-of-the-art entity linking solutions are also limited in performance [21]. Especially, linking entity mentions to entities in local PKGs since these graphs are dynamically evolving through conversations and have different identifiers from person to person [7]. Some of the works also use a representation of knowledge graphs instead of solely using text [50]. Few works explore complete entity linking solutions, i.e. both identifying entity mentions and linking them to the corresponding KG entity [2, 6, 11, 39].

Our entity linking to a local PKG problem differs from existing entity linking solutions in that existing solutions use unique identifiers for KG entities, whereas this is impossible for our local PKGs because their identifiers vary from PKG to PKG. When performing entity linking without the use of a knowledge graph, the problem can be viewed as the coreference resolution problem, which may also be ideal for mapping pronouns [5].

Coreference Resolution. Coreference resolution is concerned with grouping entity references that refer to the same entity [54]. This is especially useful in determining what the pronouns in the conversations refer to [12, 22, 23, 54].

Heuristic models are commonly used to determine coreference in the early days of coreference resolution [54]. This is performed by creating a set of rules capable of capturing structural information in a text and using these rules to decide which mentions are referencing each other [22].

As it is quite cumbersome to cover all edge cases with explicit rules, deep learning has been applied for this purpose [12, 23, 54]. There has been success with end-to-end deep learning models, such as using a bidirectional LSTM with inner-attention [23]. Furthermore, using reinforcement learning to improve current heuristic coreference modules has also produced great results [12].

Pre-trained Language Models. A language model is a probability distribution over a set of words [35]. Language models are widely used for downstream natural language processing (NLP) tasks, especially pre-trained language models [35, 43]. These are language models that are trained on a large dataset of natural language using self-supervised learning. Pre-trained language models are used in two ways for NLP tasks [53]. The first way is as a feature-based approach, where the word embeddings learned in the pre-trained language model is used as features by NLP models [53]. The second way is a fine-tuning approach, where the last layers of the pre-trained language model is tailored specifically to an NLP task [53].

Amongst the pre-trained language models, the joint knowledge embedding and language model seeks to represent knowledge graph data and language simultaneously [35]. Some works incorporate triple-based information into the text to learn a joint representation [25, 33, 43, 47]. They are similar to our problem in that a representation of both language data and external knowledge graph data is learned. It differs from our problem of linking entities as the final goal of the joint language model and knowledge embeddings is a representation of words and knowledge graph data (i.e., entities and possibly relations), which is different from our task of linking entity mentions directly to a local PKG.

3 PROBLEM FORMULATION

We first formally define the problem of identifying and populating personal entities into a PKG as the PKG Statement Linking problem. Secondly, we define the problem of modelling the relation between the personal entities and the open KG entities as the PKG Enrichment problem.

As the basis of PKG population is a Knowledge Graph (KG), we first formalise KG.

Definition 3.1 (KG)**.** We define a KG to be a graph $G = (\mathcal{E}, \mathcal{R}, \mathcal{L}, l)$:

- $\mathcal{E}$, set of entities.
- $\mathcal{R} \subseteq \mathcal{E} \times \mathcal{L} \times \mathcal{E}$, set of relations.
- $\mathcal{L}$, set of labels. It consists of names for entities and relations, which means there exists a mapping such that $l : \mathcal{E} \cup \mathcal{R} \mapsto \mathcal{L}$.

The entities in the KG can be either concrete things or abstract concepts, and for the sake of simplicity, we will interchangeably refer to them as entities.

As a prerequisite for our tasks, we have triple extraction and open KG entity linking. Triple extraction extracts triples from an utterance, i.e., a subject substring, an object substring, and the relation between them. Open KG entity linking maps each mention/substring in the textual triples to the corresponding open KG entity.

We illustrate the problem of PKG statement linking in Figure 3. The problem concerns linking subject/object mentions from the textual triples to entities in an existing PKG, if the appropriate entity exists. If the appropriate entity does not exist yet, then it should be created as a new personal entity in the PKG.

Problem Definition 1 (PKG Statement Linking)**.** Given a user PKG G_u:$(\mathcal{E}_u, \mathcal{R}_u, \mathcal{L}_u)$ for the user u, an open KG G_Ω:$(\mathcal{E}_\Omega, \mathcal{R}_\Omega, \mathcal{L}_\Omega)$, utterance q_u from the user, n textual triples $\{\langle s_x, p_x, o_x \rangle | 0 < x < n \wedge p_x \in \mathcal{R}_\Omega\}$ extracted from q_u from the triple extractor, a dialogue history $\{q_1, ..., q_{u-1}\}$ between the user u and another actor, and possibly a set of entities from G_Ω for each s_x and o_x in the textual triples from the open KG entity linker, the PKG Statement Linking problem requires to map each of s_x and o_x to a member of $\mathcal{E}_u \cup \{+_u\}$, where $+_u$ means that the node is not present but should be added to $\mathcal{E}_u$[1]. The mappings of each s_x and o_x are related by the p_x relation.

Besides linking the textual triples to an existing PKG, it is also necessary to model the relation between the PKG entities and the open KG entities through additional triples, e.g, personal entity e₄ is a snake as illustrated on **Figure 4**.

[1]Note that we cannot modify G_Ω

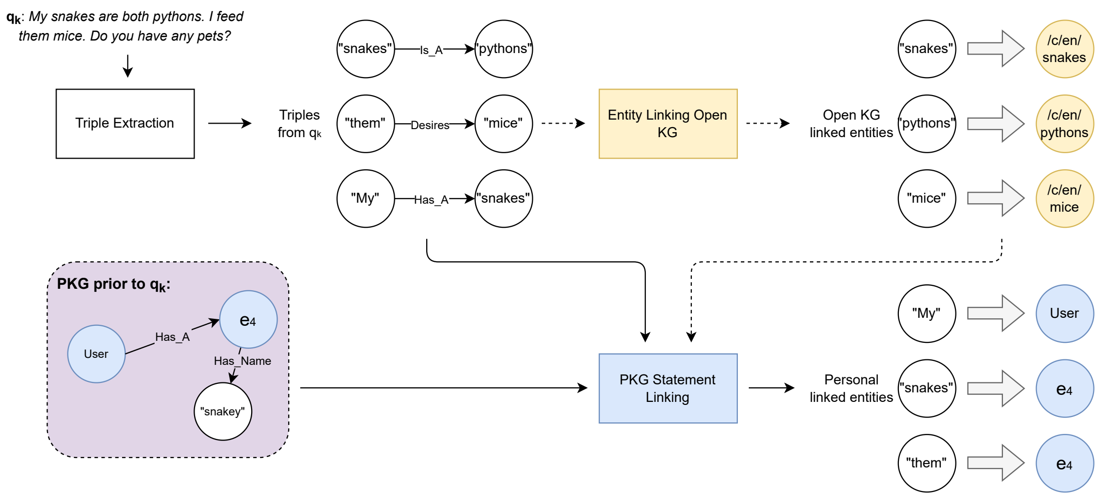


Figure 3. Given extracted textual triples, PKG statement linking concerns mapping the textual triples to entities in a PKG. Optionally, we could also provide the mapping textual triples to a open KG as input.

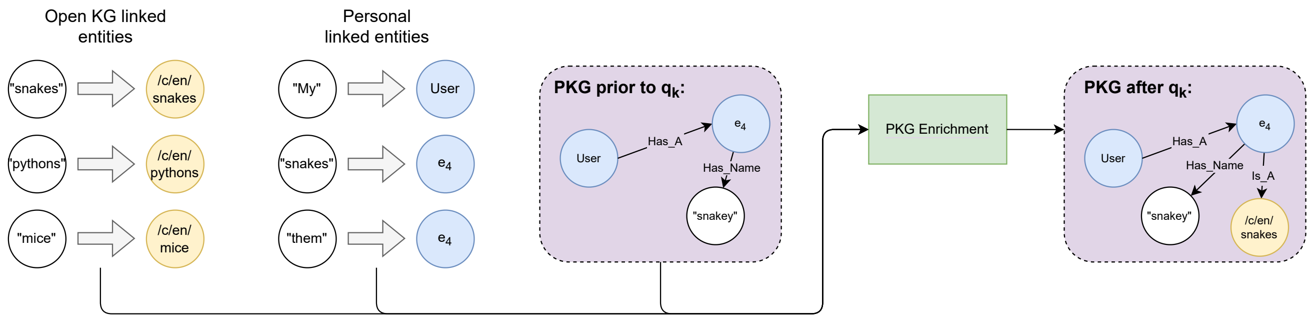


Figure 4. Illustation of the PKG Enrichment task.

These additional triples model information that is not explicitly mentioned in the conversation but can be inferred.

We could model this problem as the learning problem of predicting the relation when a subject/object mention in a textual triple is simultaneously linked to both the PKG through **Problem 1** and through Entity Linking Open KG in **Figure 3**. Nevertheless, due to the scope of this project, we will consider the relations in these triples as is_A relations.

Formally this problem is defined as the problem of *PKG Enrichment*:

Problem Definition 2 (PKG Enrichment)**.** We formulate the problem of enriching the PKG with triples, open KG entity mappings, and personal entity mappings. Given the output of **Problem 1**, i.e., the personal entity mappings and the relations between them, and the entity mappings to the open KG, the problem is to generate a new PKG that is a supergraph of the input PKG with added triples given the various inputs. These triples model the relations between the personal entities and the open KG entities.

A running example of the complete architecture with existing components along with PKG Statement Linking and PKG Enrichment is shown in **Appendix A**.

4 SURVEY ON COMMON SENSE KNOWLEDGE GRAPHS

In the PKG, we differentiate between two types of entities, namely personal and open KG entities. The open KG entities refer to entities from a Common Sense Knowledge Graph (CSKG). We, therefore, examine several CSKGs, which we present with their important characteristics to understand and evaluate the most applicable one in our use case.

4.1 ConceptNet

Speer et al. [41] present ConceptNet, which is populated by combining various common sense knowledge sources. They use 49 relations, where 7 of them are introduced as symmetric between the entities. The CSKG contains ≈24 mio. entities and ≈28 mio. triples. The implementation of ConceptNet is available on Github, and they have also provided an API to it. The newest version available of ConceptNet is 5.7, but ConceptNet 5.5 is the latest CSKG published at a conference. ConceptNet 5.7 is larger and more extensive compared to 5.5 and uses 50 relations.

4.2 ATOMIC

Sap et al. [36] present another popular CSKG from 2018 named ATOMIC, the abbreviation of an Atlas of Machine Commonsense. It contains 9 relations over around 300.000 nodes and 730.000 edges. This graph is created by first gathering a large set of common event phrases, and thereafter crowd-sourcing annotations to further understand the natural behaviour around this event. In comparison to ConceptNet, ATOMIC focuses more on chains of events with social commonsense behind. This means that there is an overlap between the two CSKGs, but they each provide important information not captured by the other.

4.3 LM4KG

Omeliyanenko et al. [32] presents LM4KG, which is an abbreviation of Language Model for Knowledge Graph. They enhance CSKGs such as ConceptNet using a language model. The intention is to assign weights to the different nodes and then adjust these weights using a language model. They propose a technique called Refined Edge Weighting that takes the existing KG and generates sentences from them, which they use as a language model to calculate perplexity. The perplexities is then used to adjust the assigned weights. The implementation for applying this technique on the dataset is also available on Github, making it possible to enhance any CSKG.

4.4 CSKG1

Ilievski et al. [19] present the CSKG1, abbreviation of *CommonSense Knowledge Graph*, from 2021. This graph is gathered based on a variety of different knowledge sources to create an extensive KG capturing common sense information. One of these knowledge sources is existing CSKGs, e.g., ATOMIC and ConceptNet 5.7. CSKG1 uses 58 relations, ≈2

Table 1. An overview of the mentioned CSKGs. The statistics regarding the different CSKGs are collected from either their respective papers or through analysing the CSKG.

Name	Year	Relations	Entities	Edges	Applicable
ConceptNet 5.5	2017	49	24.032.766	28.860.810	Yes
ConceptNet 5.7	2019	50	28.370.083	34.074.916	Yes
ATOMIC	2018	9	304.909	732.723	No, not graph format
LM4KG	2020	-	-	-	No, graph not published yet
CSKG1	2021	58	2.087.833	5.748.405	Yes
Wikidata-CS	2020	15	66.192	101.770	No, it doesn't have sufficient entities

mio. entities, and ≈6 mio. edges. CSKG1 is publicly available on Github[2].

4.5 Wikidata-CS

Ilievski et al. [18] present a small CSKG called Wikidata-CS, which is extracted as a subgraph containing the common-sense facts of the Wikidata KG. This CSKG uses 15 relations and contains ≈66.000 entities and ≈101.000 edges.

4.6 Evaluation

After investigating the mentioned CSKGs, we observe that CSKG1 and ConceptNet are likely the best fit for our problem. We present the result of our investigation in **Table 1**.

The column 'Applicable' describes whether we can use the CSKGs to model the conversation. To this end, we apply the relations to a small set of randomly selected conversations to see whether the CSKGs can capture all important personal information. Furthermore, we use Gestalt pattern matching to determine whether there are candidate entities for each recognised entity mention by Spacy entity recogniser [40] in the selected conversations. In this candidate entity search, we observe that CSKG1 and ConceptNet are both capable of finding candidates for all entity mentions in the sample. Nevertheless, we find that the subset of entities used by CSKG1 to find candidates belongs to ConceptNet. Furthermore, as ConcepNet has additional entities, we have decided to choose it as our CSKG. Additionally, we find the relations more descriptive in ConceptNet 5.7 compared to CSKG1.

5 PERSONAL KNOWLEDGE GRAPH DATASET

To properly construct and evaluate PKG population, we need a conversational dataset with textual triple annotations and entity annotations on each textual triple. We need two forms of entity annotation: one referring to an open KG and the other to a local, conversation-specific PKG.

5.1 Pre-Processing

Knowledge Sources. As a source of conversations, we use the Persona-chat dataset [55]. As a CSKG, we will use ConceptNet 5.7 as concluded in **subsection 4.6**.

Conversation Ranking. As an initial step, we rank the conversations based on a set of patterns to ensure that the conversations contain the personal information usable for our problem definition. We use the patterns in **Appendix B**. We rank them by assigning scores based on the frequency of the matched patterns. We start the annotation process on the conversations which have the highest score.

5.2 Annotation Phases

Triple Annotations. We first developed a triple annotation interface with the library Prodigy[3]. We then annotated 100 conversations with entity mentions and the relation between each pair of them in the conversation. Our set of relations and their specification are presented in **Table 2**. We show an example of the annotation website in **Appendix C**.

ConceptNet Entity Annotations. Besides textual triple annotations, we need entity linking annotations to ConceptNet. We have also developed an interface for annotating entity mappings to ConceptNet with the Prodigy library. We have sought external assistance to collect these annotations. We first create a pool of candidate entities from ConceptNet for each subject and object mention in the conversations, which the external annotators can choose from. We find the candidate entities from ConceptNet by using string-wise similarity[4]. To the annotators, we display an utterance from a conversation and a single entity mention in the utterance along with the candidate entities. The annotators will then be given the option to choose from the candidates. If none of the candidates correspond to the entity mention, they will also have the option to choose that the correct entity is missing. They also have the option to select that they don't have enough context to deem, which entity is the correct. This annotation system is also depicted in **Figure 5**.

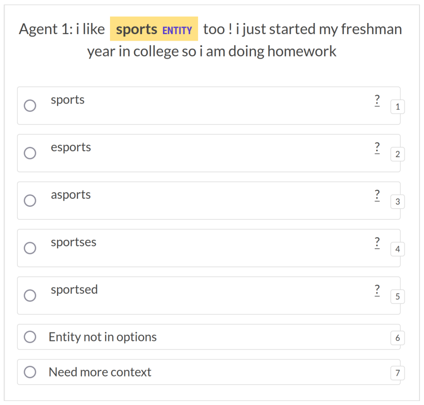
[2]https://github.com/usc-isi-i2/cskg

[3]Prodigy is an annotation library for NLP tasks

[4]Python's difflib library provides this functionality. It uses Gestalt Pattern Matching as the similarity measure

Table 2. Relations used in our manual triple annotation phase

Relation Name	Description
At_Location [41]	The location of an entity
Desires	Covers the preference_relation in Table 8
Has_A [41]	Covers the owner_relation in Table 8
Has_Property [41]	The property of an entity. The value should also be an entity
Is_A [41]	is used when an entity is an instance of another entity
Part_Of [41]	The entity is a component of the other entity
Aversion	Covers the aversion_relation in Table 8
Additional Relations	
Has_Name	The name of an entity. The object will be a string literal.
Has_Age	The age of an entity The object will be a string literal.(c/en/of_age from ConceptNet)
Has_Gender	The gender of a Person entity. The object will be an entity.(c/en/gendered from ConceptNet)
School_Status	Entities related to the school status of an entity. The object will be an entity.(/c/en/attend_school from ConceptNet)
Job_Status	Entities related to the job status of an entity. The object will be an entity. (c/en/occupation from ConceptNet)
Has_Value	The value of an entity. The value should be a string literal


Agent 1: i like sports ENTITY too ! i just started my freshman year in college so i am doing homework

- sports 1
- esports 2
- asports 3
- sportses 4
- sportsed 5
- Entity not in options 6
- Need more context 7

Figure 5. An example of the ConceptNet entity annotation interface.

Personal entities. We also require personal entity annotations, i.e., annotations indicating which of the entity mentions refer to the same personal entity. The relation annotations interface from the Prodigy library is modified for this purpose. It should also be possible to mark relations between the subject and object mentions from the textual triple annotations, and the only available relation should be a *COREF* relation. We show an example of this annotation website in **Appendix C**. After the annotators have marked the personal entities in a conversation, each personal entity mention is assigned an identifier, unique to each personal entity.

5.3 Post-Processing

After collecting the annotated data, we have three separate files of each of the annotation types. We present samples of the annotated data in **Appendix D**.

For ease of use in our experiments, we gather the three separate files into a single data source. To accomplish this, we divide each conversation in our personal entity annotations and triple annotations into utterances, and readjust the start and end indices of the entity mentions to reflect the utterance instead of the entire conversation. We also assign personal entity identifiers to the entity mentions by first assigning new identifiers to the subject of the COREF relations of the personal entity annotations data. Next, for objects of the COREF relations, we add the personal identifier of the subject. Lastly, for entity mentions not part of any COREF relations, we add new personal identifiers. In the final iteration of the dataset, we combine the annotation data of textual triples and the entity mentions, i.e., the start and end indices of an entity mention, personal entity identifier, and ConceptNet entity mapping. We present a sample of this dataset in **Appendix E** to gain a better insight into the data.

5.4 Dataset Statistics

After conducting the three different annotation phases, we have gathered a significant dataset for the purpose of PKG population. The created dataset consists of three different files with triple annotations, personal entity annotations, and ConceptNet entity annotations. The statistics of this dataset is as seen on **Table 3**.

# of Conversations	100
# of Utterances	1674
# of Triples	2204
# of CSKG Entities	2263
# of Unique CSKG Entities	836

Table 3. The statistics of our dataset

6 OUR METHODOLOGY

We present our approach to solving **Problem 1** as a personal entity linker. We propose this personal entity linker as a two-step architecture consisting of a personal entity classifier (PEC) and a personal entity disambiguator (PED). PEC determines whether a given entity mention in the input is an existing personal entity in a PKG. Given the output of the PEC, the PED will then link the entity mention to a personal entity.

6.1 Transformer

Vaswani et al. [45] present the *transformer* architecture, which is a sequence-to-sequence model based on attention mechanisms. An overview of the transformer architecture is shown in **Figure 6**.

Transformers use the encoder-decoder structure. The architecture consists of an input embedding layer, an encoder stack and a decoder stack, where each stack contains X encoders and decoders. The input embedding layer consist of token embeddings and position embeddings that are element-wise added. It additionally contains a fully-connected layer with softmax activation function after the decoder stack. An encoder consist of two layers, where the first is a multi-head self-attention layer, and the second is a fully connected feed-forward layer. Around each of the layers in the encoder, residual connection and layer normalisation is applied. The decoders contain an additional layer of attention, which applies attention over the output of the final encoder in the encoder stack. The encoder stack constructs a context representation Z from a source sequence $X = (x_1, x_2, ..., x_n)$, which the decoder layers utilises to generate the target sequence $Y = (y_1, y_2, ..., y_m)$.

The attention layer can be viewed as mapping a query and a set of key-value pairs to an output, which is a weighted addition of the values. The weights are determined by the similarity between the query and keys. It works by first transforming the input into three matrices Q and K of dimension d_k, and the third matrix V of dimension d_v.

$$Q, K, V = XW^Q,\ XW^K,\ XW^V$$

, where W^Q, W^K, W^V refer to the learned parameters.

Then, the attention is calculated as follows:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

This operation is called *scaled dot-product attention*. Furthermore, instead of calculating a single attention, they use *multi-head attention*. This works by performing attention a number of times in parallel that are then concatenated and finally projected.

Masked Transformer Encoder. The masked transformer encoder is based on the previously described transformer theory. It calculates attention using a similar algorithm, but includes a masking matrix.

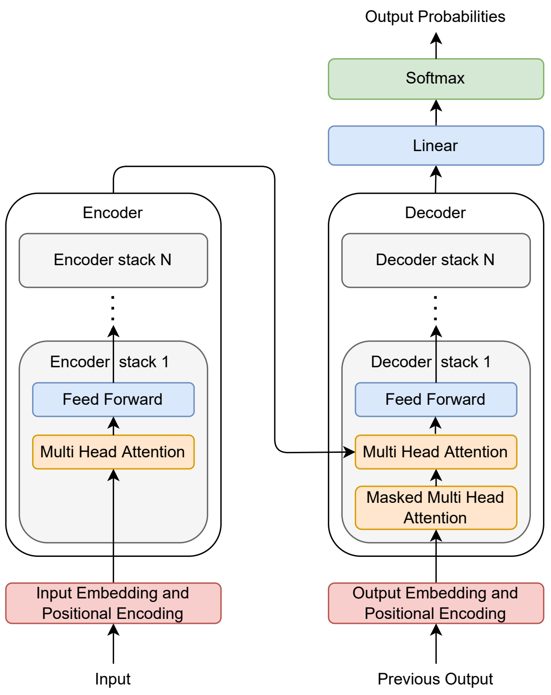


Figure 6. Transformer architecture. Figure inspired by Vaswani et al. [45]

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}} + \mathbf{M})V$$

, where $M \in \mathbb{R}^{n \times n}$ refers to the masking matrix.

Each element $M_{i,j}$ indicates whether x_i can access x_j and vice versa. If they can access each other, the value will be 0, otherwise the negative infinity will be used.

6.2 Personal Entity Classifier

Our personal entity classifier is a modified version of the mentioned transformer architecture, with the addition of a new representation of the inputs described in **Problem 1**, which is inspired by Liu et al. [25] and Sun et al. [43]. This input representation is the concatenation of a PKG and a representation, which we refer to as the Utterance Relation Graph (URG). We also modify the transformer's embedding and encoding layers to accommodate this input representation.

Utterance Relation Graph. In our **Problem 1**, we define the inputs to be natural language data (i.e., utterances), knowledge graph data (i.e., a PKG and open KG entities), and a mix of these in the form of textual triples, in which subjects and objects are natural language and relations are from a knowledge graph. As our data is natural language and knowledge graph data, it makes sense to draw inspiration from joint language modelling and knowledge embedding, which

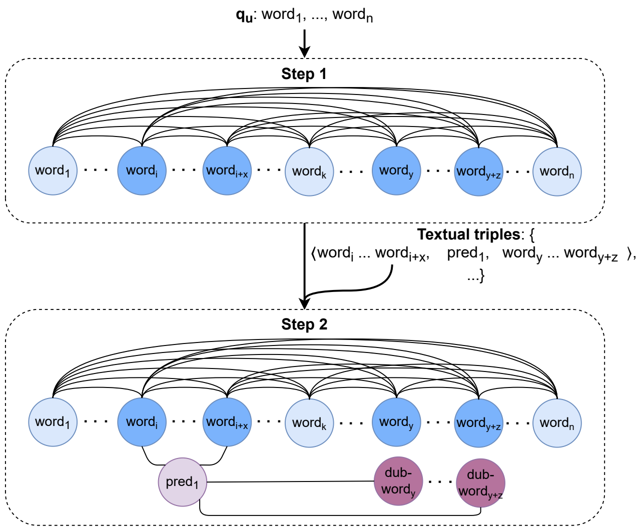

q_u: word$_1$, ..., word$_n$
Step 1
word$_1$
word$_i$
word$_{i+x}$
word$_k$
word$_y$
word$_{y+z}$
word$_n$
Textual triples: { ⟨word$_i$ ... word$_{i+x}$, pred$_1$, word$_y$ ... word$_{y+z}$ ⟩, ...}
Step 2
pred$_1$
dub-word$_y$
dub-word$_{y+z}$

Figure 7. Dataflow of the URG construction steps.

specialises on combining natural language and knowledge graph representations.

The intuition behind our URG is based on the observation that the input to the transformer is essentially considered as a fully connected graph of each $x_i \in X$ in the transformer architecture due to the self-attention mechanism. This observation was made by Sun et al. [43]. With this observation in mind, we can extend this data representation with the additional information available (i.e., textual triples and possibly ConceptNet mappings). This results in our input data representation, URG.

We build the URG by first dividing the input utterance into word segments, $q_u = \{\text{word}_1, \ldots, \text{word}_n\}$. Each word segment is considered a word node, and together they form a fully connected graph. This is shown as Step 1 in **Figure 7**. We next add textual triple information to this graph, where a textual triple is $\langle \text{word}_i \ldots \text{word}_{i+x},\ \text{pred},\ \text{word}_y, \ldots, \text{word}_{y+z} \rangle | x >= 0 \land y >= 0$, in which the subject and object are entity mentions from q_u. We first add a relation node for each *pred* in textual triples, and edges from each word $\in \text{word}_i, \ldots, \text{word}_{i+x}$ to the relation node. We also duplicate the object entity mentions $\text{word}_y, \ldots, \text{word}_{y+z}$ and add edges to the relation node. This is shown as Step 2 in the figure, where this process is shown for a single textual triple. We consider the word nodes connected to relation nodes as a special type of word nodes, namely entity mention nodes. This process is also illustrated in **Figure 7**. In total, we distinguish between word nodes, relation nodes, and entity mention nodes in our URG. Additionally, for the particular entity mention, which we would like to classify as present in the input PKG or not, we encode as the interest node.

Personal Knowledge Graph Representation. Aside from the URG, we must also provide our classifier with a representation of the PKG. In the PKG, we distinguish between personal entities, relations, and ConceptNet entities. Because we use the same relations in both, we keep the same representation of relations as in the URG. We keep a local identifier per PKG for personal entities. The PKG representation, we will provide to our model, will be an undirected graph consisting of a set of personal entities, a set of ConceptNet entities from the PKG, and a set of predicates from the set of triples of the PKG. The edges in this graph will be added according to triples, i.e., for the triple $\langle e_1,\ p_1,\ e_2 \rangle$, we add the edges (e_1, p_1) and (p_1, e_2).

Embedding Layer. We require a representation of both the URG and the PKG in the embedding layer of our transformer model. The input embedding, $X \in \mathbb{R}^{n \times d}$, is the addition of three embeddings; token embedding, type embedding, and position embedding. n denotes the total amount of nodes in the URG and PKG, and d is the dimension of each node.

As token embedding, we create one-hot encoding of personal entities in PKG and keep lookup tables for words and ConcepNet entities and a joint lookup table for relations in PKG and URG. We keep one lookup table of the relations from both URG and PKG since the relation nodes are common for both. For the words, we use the Byte-Pair Encoding [38] as RoBERTa [26] to transform sequence into subword units. For the ConceptNet entities and our relations, the embeddings will be learned directly. The token embeddings are calculated by concatenating the embeddings of PKG entities, relations, words, and Concepnet entities. This results in a token embedding of size $T \in \mathbb{R}^{n \times d} | n = (|CLS| + |pkg\ entities| + |pkg\ relations| + |SEP| + |words| + |triple\ predicates| + |object\ words|)$, where CLS denotes the special classification token used for classification tasks in language models, and SEP denotes the separation token.

As we have different type of nodes in our input, we use type embedding to represent the different types of nodes, i.e., word node, relation node, entity mention node, interest node, personal entity node, and possibly CSKG entity node.

For the position embedding, we use soft-position embedding similar to Liu et al. [25] and Sun et al. [43]. **Appendix F** shows the intuition behind the embedding layer.

Encoder Layer. We use the masked transformer encoder, as presented in **subsection 6.1**, to encode the URG.

We use the mask specified by:

$$M_{ij} = \begin{cases} 0 & \text{if } x_i \text{ and } x_j \text{ both in URG or PKG and connected} \\ -inf & \text{if } x_i \text{ and } x_j \text{ are disconnected} \\ -inf & \text{if } x_i \text{ in URG and } x_j \text{ in PKG} \end{cases}$$

Output Layer. On top of the transformer model, we also add a classification head, which is tailored toward predicting whether the entity mention of interest is present in the input PKG or not. This classification head consist of two fully connected layers, which takes the output of the transformer as the input. The output of the classification head will be a value which we threshold at 0.5. If the value is greater than 0.5, we consider it as the entity mention being present in the input PKG, and otherwise it is not.

6.3 Personal Entity Disambiguator

The Personal Entity Disambiguator (PED) links an entity mention to an entity in a local PKG using a heuristic-based method. This method is dependent on whether the mention is classified as a new or existing entity by the PEC component. The approach is described in **Algorithm 1**.

Algorithm 1 Pseudo code for Personal Entity Disambiguator

```
PED (mention, utterance, text_triples, dialogue, lookup, PKG):
begin
  if mention is personal pronoun then
    return id of user node
  else if mention is pronoun then
    x ← coreference(mention, dialogue + utterance)
    match ← string-similarity(x, lookup)
    add (x, match) to lookup
    return match
  else
    PEC_pred ← PEC(utterance, text_triples, PKG, mention)
    if PEC_pred is new then
      create new id for mention
      add (mention, new id) mapping to lookup
      return new id
    else if PEC_pred is not new then
      match ← string-similarity(mention, lookup)
      add (mention, match) to lookup
      return match
    end if
  end if
end
```

In PED, we keep a lookup table, which maps previous entity mentions to personal entity identifiers. It is also updated after each utterance in the conversation to reflect the personal entity ids in the PKG. PED works by first checking whether the mention is a personal pronoun as we know personal pronouns are references to the user. If it on the other hand is a non-personal pronoun, we use coreference resolution to check the closest entity mention in the dialogue history and use this mention to find the personal entity id in the lookup table by comparing the mention to the keys of the lookup table.

If it is not a pronoun, we use the PEC classifier to determine whether it already exists in the PKG. If that is the case, we compare the entity mention to the keys of the lookup table as with we with the pronouns. We also update the lookup table with the entity mention and the personal entity id. If the PEC classifier predicts that it is a new entity, we add a new entry to the lookup table with mention and a new identifier.

We do presume triples with certain relations (i.e., *hasValue*, *hasName* and *hasAge*) beforehand to have a literal as object instead of entity, which means no linking is required.

7 EXPERIMENTS

We present our experiments in this section. As presented in **section 6**, our personal entity linker consists of two main components, which are namely the PEC and PED. We will show the results of classification using PEC and the results of the entity mentions linked to the PKG using the PED.

7.1 Experimental Setup

Our conversations consist of two actors, where we have annotations specific to each actor. We treat each conversation as two conversations with each of the actors in focus. As one of the core components of our personal entity linker is the PEC, which requires training a model, we split our dataset into training, validation, and test sets, with a 0.6/0.2/0.2 split. We present the statistics of each split in **Table 4**.

Table 4. Statistics of the different sets

	Train	**Validation**	**Test**
Conversations	120	40	40
Utterances	1040	302	332
Triples	1312	389	419
Distinct Personal entities	1359	410	435
Unique Concept-Net entities	581	216	228

Initial PKG configuration. We examined the amount of reoccurring personal entities in the conversations. We present the frequency of each specific amount of reoccurring personal entities in the dataset in **Figure 8**.

From **Figure 8**, we notice that the majority of the personal entities only occur a single time in the conversations. This is likely due to the conversations being from PersonaChat, where we observe a rapid shift in the topics of the conversations. As a result of the excessive volume of single occurrence entities, the dataset becomes highly imbalanced such that classifying the entities as new ends up being more significant than linking to existing entities in the local PKG. Furthermore, this distribution is also not ideal for the PEC as a balanced dataset is important for training a model. To overcome this imbalance, we provide each utterance in 50%

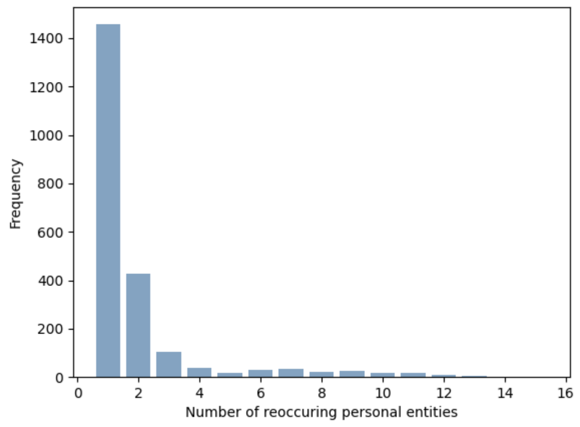


Figure 8. The frequency distribution of the amount of times the personal entities reoccur in a conversation.

of the conversations with an initial PKG constructable from the entire conversation.

7.1.1 Personal Entity Classifier.

Implementation Details. As our PEC is a deep learning model, we train it. We use RoBERTa-base as an initialisation of PEC. As the objective function, we use mean squared error. As the optimizer, we use AdamW with β_1 of 0.9, β_2 of 0.999, and learning rate of $5e^{-5}$. We also normalise and batch the dataset of size 16, where we ensure the same amount of positive and negative samples during the training. We use an NVIDIA GeForce GTX 1060 6GB GPU and train the model for approximately 2 hours.

URG and PKG construction. Before training the PEC, we prepare the URG and PKG for the model. We construct a PKG for each utterance q_i in the conversations by gathering the textual triples from the previous utterances $q_x|0 \leq x < i$ and replacing the entity mentions with their respective personal identifiers. If the entity mentions em_i in the textual triples are also mapped to CSKG entities $CSKG_i$, we additionally add the triple $\langle pers\_id_i, is\_A, CSKG_i \rangle$ to take **Problem 2** into account.

Next, we prepare input embeddings for the PEC. For each input utterance in the conversations, we implement and calculate the embeddings described in the Embedding layer of **subsection 6.2**. Furthermore, for each entity mention in the textual triples, we duplicate the constructed URG, and the token embedding of the entity mention is changed to represent the interest node.

The mask used in the masked encoder layer is also pre-computed when we construct the graph containing both the PKG and URG, i.e., the entire input to the PEC.

We also pad the input embedding such that the first n_pkg_ents elements represents personal entities from the PKG, while the next n_pkg_cskg elements reflect references to CSKG entities in the PKG. Likewise, the next n_pkg_rel, n_words, $n\_textual\_rel$, and $n\_obj\_mention$ represent PKG predicates, word embedding of the utterance, relations from textual triples, and the duplicated object mentions of the textual triples, respectively.

7.2 Reproducibility

We aim for our work to be as reproducible as possible. To this end, we used Docker to containerize our personal entity linker to provide full reproducibility of our model for evaluation. We've also picked a certain seed for the model to avoid it changing due to the randomness. Furthermore, we also make our dataset available.

7.3 Personal Entity Classification

For experiments on our solution, we will present the performance of our PEC model. We will additionally compare it against other baseline classifiers, where the input will be a vector x_i for each sample, where x_i consists of the utterance tokenised using RoBERTa [26] concatenated with an entity mention.

Support-Vector Machine. Within supervised machine learning, a well-known model is the support-vector machine. They are built on the idea of having a support vector sectioning the feature space using the training data such that it is possible to classify new examples by inserting them into the feature space.[13]

Neural Networks. A neural network is a deep learning architecture, which has become quite popular in recent years due to its high performance in machine learning and pattern recognition. These networks consist of layers of learnable weights, which are used to predict [37].

7.3.1 Results. We have tested different configurations of our PEC and the baseline models. We present the precision, recall, and F1-score on the training, validation, and test sets of our PEC along with the baseline models in **Table 5**. In the table, the support-vector machine achieves the best result on the training dataset with the precision, recall, and F1 being 0.99. These measures are depicted as the P, R, and F1 columns. For the validation dataset (the Val P, Val R & Val F1 columns) and the test dataset (the Test P, Test R & Test F1 columns), our PEC achieves highest results, where it gets 0.83-0.84 on the validation set and 0.74-0.77 on the test set. It is clear to see that PEC outperforms the neural network and support-vector machine on the unseen data (i.e, validation and test datasets). On the baseline models, we also observe a greater extent of overfitting, while this is barely noticeable in PEC, in that the difference between the training performance and the validation/test performance is small for PEC.

Table 5. Results of classifying the personal entities. P stands for precision, R for recall, and F1 for F1-score. The first three columns depict the performance on the training dataset. The next three columns depict the performance on validation dataset. The last three are the results on the test dataset

	P	**R**	**F1**	**Val P**	**Val R**	**Val F1**	**Test P**	**Test R**	**Test F1**
PEC	0.77	0.72	0.72	**0.84**	**0.84**	**0.83**	**0.77**	**0.73**	**0.74**
Neural Network	0.79	0.88	0.82	0.47	0.67	0.54	0.48	0.53	0.49
Support-vector machine	**0.99**	**0.99**	**0.99**	0.50	0.46	0.48	0.48	0.49	0.48

7.4 Personal Entity Linker

We also evaluate our personal entity linker on the validation and test split of the dataset. Our personal entity linker uses string similarity measure, and an external coreference solution for pronouns that do not refer to the user. As the string similarity measure, we use Gestalt pattern matching. We experiment with two coreference solutions, namely Stanford CoreNLP and NeuralCoref.

StanfordCore NLP Coreference Resolution [22]. StanfordCore NLP's coreference resolution approach is based on deterministic rules. Hand-tuned weights are used in this rule-based Coreference Resolution, which works well with a smaller number of utterances at a time.

NeuralCoref. Using Spacy and the Neural Mention-Ranking model presented by Clark and Manning [12], NeuralCoref is able to extract coreference clusters. The neural mention-ranking model ranks entity mentions for coreference compatibility using a feedforward neural network.

7.4.1 Results. We consider the output of the personal entity linker as a multi-class classification problem and compare it against our personal entity identifiers in our annotated dataset. To make this comparison, we modify our personal entity linker to assign the personal entity identifiers from the ground truth when it needs to create a new id for a personal entity. We use macro precision, recall, and F1-score to evaluate the results of personal entity linker. The results are presented in **Table 6**, where we evaluate the model with the two different corefence modules given the classifications from PEC. The last row show the evaluation of the PED given the ground truth personal entity classification. Of our personal entity linker with an external coreference, the linker with NeuralCoref performs better than StanfordCoreNLP with an improvement on 0.01 across all metrics with 0.91 0.86 and 0.87 as precision, recall and F1-score, respectively.

Table 6. Results of disambiguating the personal entities. The P, R and F1 columns are the macro precision, recall and F1-score, respectively.

	P	**R**	**F1**
PED w. NeuralCoref	**0.91**	**0.86**	**0.87**
PED w. StanfordCoreNLP	0.90	0.85	0.86
PED w. perfect PEC	0.99	0.998	0.996

8 DISCUSSION

In this work, we have worked with the problem of PKG statement linking. As a result, we developed our own personal entity linker. We will expand on our considerations for these outcomes in more detail.

8.1 Personal Entity Classifier

We proposed the PEC as the core of our personal entity linker with the assumption that if we know whether a specific entity mention is present in the input PKG, then we can relatively easily link to the specific entity in the input PKG. This is also supported by the information shown in **Table 6**, where the PED with perfect classifications of whether a mention references an entity in the input PKG performs almost flawlessly. Even though our architecture seems to outperform the baseline models by $\approx$ 35-42% in F1-score, we still find that the architecture has room for improvements.

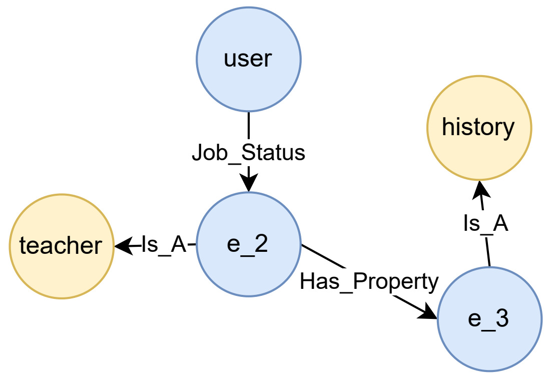
The PKG representation for the PEC is one area that could be improved, more specifically the personal entity representation. In the current PEC, we provide one hot encodings of the personal entities in the PKG. This essentially works as a local, conversation-specific lookup table. An alternative could be to use node embeddings like the work by Grover and Leskovec [15].

Another area we could potentially improve upon, is the URG, which we could extend with ConceptNet mappings. Given that there exists a mapping between an entity mention node in the URG and a ConceptNet entity, we could add the ConceptNet entity to the entity mention node in the URG. This could potentially provide the URG with more contextual information. We show an example of this extended URG in **Appendix G**.

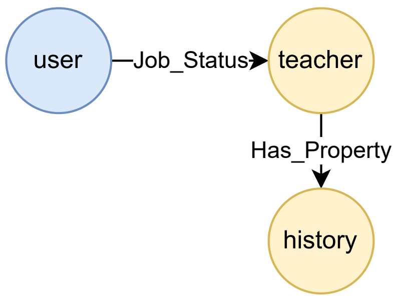
Additionally, we could improve the architecture by adding more explicit connections between the URG and PKG since they are considered disconnected graphs in the input. We could add edges between the ConcepNet entities and relations in the PKG to the URG if we observe the same relation/Conceptnet entity in the input.

8.2 Personal Entity Disambiguator

In our experiments with PED, we observe a minimal difference between using the two different coreference solutions. This indicates that their use has no observable effect. We investigated how often pronouns that do not refer to the user are present in the validation and test datasets and find



(a) PKG populated with architecture from introduction extended with the personal entity linker.



(b) PKG populated with the architecture from the introduction.

Figure 9. A snippet of the PKG constructable from conversation with id 234 in our PKG population dataset

that only 9 pronouns are present. From these, we looked into how often the coreference resolution is capable of finding a mention in the dialogue history to link to. Of the 9 pronouns, Stanford CoreNLP is capable of finding an entity mention from the dialogue history 5 out of the 9 pronouns, while NeuralCoref is only able to accomplish this with 3 of them.

We do not expect the current personal entity linker to perform well in a practical setting, because we observe from the precision measure of PEC that 2-3 out of every ten entity mentions classified as existent does not actually exist in the PKG. This would mean that we likely link the mention to the wrong personal entity instead of representing it as a new personal entity. Meanwhile, a recall of 0.73-0.84 means that we add new entities to the PKG for every 1-3 mentions out of ten. This issue seem less accentuated in the results of PED with F1-score of 0.87, where the score indicates the entity mentions that are linked correctly. We also consider the user entity, which we due to the heuristic-based method almost always links correctly.

8.3 Comparison of populated PKGs

We compare the PKG population architecture with and without the personal entity linker. We show the PKG population architecture without the personal entity linking in 9(b) and with it in 9(a). It is clear that the PKG populated with personal entities is superior to the one without it in the sense that we do not add facts about entities that are not always true, e.g., in 9(b) every 'teacher' has the property 'history' but with the introduction of personal entities in 9(a) it is possible for only specific teachers to have the property 'history' rather than all teachers. Nevertheless, the personal entity linker also has weaknesses as we see with the inclusion of the personal entity e_3, where it might make more sense to omit the personal entity and add the triple $\langle e_2, \mathit{Has\_Property}, \mathit{history} \rangle$ directly. This indicates that with the personal entity linker, the **Problem 2** PKG enrichment is a relevant problem.

9 CONCLUSION

In this work, we have expanded the research within the relatively new field of PKG population through creation of a specialised dataset and exploration of the entirely new field of personal entity linking to local PKGs.

After assessing a PKG population architecture of existing solutions, we contribute with the problems of PKG statement linking and PKG enrichment as important elements in a PKG population architecture.

We have extended the PKG population architecture of existing solutions with our solution for the PKG statement linking problem, the personal entity linker.

We evaluate our personal entity linker by comparing PEC to baseline models, where it outperforms them significantly. Additionally our experiment on PED shows that PEC is the critical part of the personal entity linker, where an improved PEC should have a big impact on the performance of the personal entity linker. Even though it performs well, we find that it is insufficient in a practical scenario.

Furthermore, to evaluate and train PKG population architectures, we contribute with a dataset containing the essential annotations. We developed annotation websites to collect the three distinct annotations used for this dataset, which we make available such that they can be used in future efforts to enhance the dataset.

Through this work, we were able to uncover more about the topic of PKG population by exploring personal entity linking and provide a solution for it. We strongly encourage more research into how to organise personal information for conversational agents using knowledge graphs, since this has the potential to provide personalised and relevant conversations.

10 FUTURE WORKS

We present possible modifications to our personal entity linker and possible future experiments. Finally, we discuss future directions in this field.

10.1 Dataset Augmentation

Our dataset contains 100 annotated conversations. With deep learning, it is important to have sufficient data to train the model on. To extend the dataset, we could experiment with different data augmentation strategies and verify whether the augmented data leads to an improved model.

From the previous semester, we investigated the state-of-the-art triple extractor [42] and entity linker [40]. We could use these implementations to collect a distantly supervised dataset by training the implementation on our current dataset. Next, we could use the distantly supervised dataset to pre-train our model, and then fine-tune on our manually collected dataset.

Another way to use the distantly supervised dataset could be to learn explicit representations of the CSKG entities and the relations in our input data. Currently, our PEC learns an implicit representation through the transformer model. As an alternative, we could use the masked language model objective function introduced by Devlin et al. [14] to learn an explicit representation of the CSKG entities and relations, and next use this in a fine-tuning phase. This would not only provide a better CSKG entity and relation representation, but we would also get a language model for conversations as it would update our word representation as well.

10.2 End-to-End Personal Entity Linking

We also considered having a multi-class classifier, which directly links the entity mentions to the PKG entities. Our idea is that because of the small number of entities in a PKG, it should be possible to modify the output layer of the PEC to directly link to a PKG entity. The output layer would then be a vector with dimension of the amount of nodes in the URG and PKG incremented by 1, i.e., if the input is $X_{in} \in \mathcal{R}^{n \times d}$, then the output would be $X_{out} \in \mathcal{R}^{(n+1)}$. We would then optimise the output to represent, which PKG entity is the correct one given the interest node (entity mention of interest) in the URG. E.g., if the x'th element in X_{in} represents the PKG entity, which the entity mention of interest should be mapped to, then the x'th element of X_{out} would be 1 and the rest 0. If on the other hand the entity mention of interest is a new PKG entity, then the $(n + 1)$'th element in X_{out} should be 1 and the rest 0.

Though this method has the potential to produce good results, we will leave it to future research to investigate. However, we do provide the source code that should be relatively easily extended to test this.

10.3 PKG in Conversational Agents

We have created a critical component of the PKG population, which is entity linking to these dynamically-evolving PKGs. This allows us to link information to personal entities, and thereby capture the personal information more accurately. This also adds the possibility of capturing several distinct personal entities, such as dogs, as separate personal entities rather than a generic concept. This is important as we otherwise would confuse information about a concept with a user's own entity. This is an important addition to the PKG population, but there is still a long way to go before these PKGs may be used in conversational agents.

KGs are already used in CAs to provide them with information, according to studies [29, 31]. Even while we can apply part of this to the challenge of using the PKGs in CAs, there is still a distinction between KGs and the smaller PKGs. To properly use the PKG, the CA should be able to navigate the PKG to the open KG and provide a response based on this. This would necessitate a plan for when to use personal information and how to traverse the graph.

ACKNOWLEDGMENTS

We would like to thank and express our gratitude to our supervisors, professor Katja Hose and assistant professor Matteo Lissandrini, for their insightful comments and assistance in this project. We would also like to thank everyone who assisted with annotating the dataset.

REFERENCES

[1] Eleni Adamopoulou and Lefteris Moussiades. 2020. An Overview of Chatbot Technology. In *Artificial Intelligence Applications and Innovations - 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5-7, 2020, Proceedings, Part II (IFIP Advances in Information and Communication Technology, Vol. 584)*, Ilias Maglogiannis, Lazaros Iliadis, and Elias Pimenidis (Eds.). Springer, Neos Marmaras, Greece, 373–383. https://doi.org/10.1007/978-3-030-49186-4_31

[2] Vipul Agarwal, Omar Zia Khan, and Ruhi Sarikaya. 2017. Remembering what you said: Semantic personalized memory for personal digital assistants. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017, New Orleans, LA, USA, March 5-9, 2017*. IEEE, New Orleans, LA, USA, 5835–5839. https://doi.org/10.1109/ICASSP.2017.7953275

[3] Rami Al-Rfou, Marc Pickett, Javier Snaider, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2016. Conversational Contextual Cues: The Case of Personalization and History for Response Ranking. *CoRR* abs/1606.00372 (2016). arXiv:1606.00372 http://arxiv.org/abs/1606.00372

[4] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging Linguistic Structure For Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics, Beijing, China, 344–354. https://doi.org/10.3115/v1/p15-1034

[5] Amit Bagga and Breck Baldwin. 1998. Entity-Based Cross-Document Coreferencing Using the Vector Space Model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*

and 17th International Conference on Computational Linguistics - Volume 1 (Montreal, Quebec, Canada) *(ACL '98/COLING '98)*. Association for Computational Linguistics, USA, 79–85. https://doi.org/10.3115/980845.980859

[6] Jiaxin Bai, Hongming Zhang, Yangqiu Song, and Kun Xu. 2021. Joint Coreference Resolution and Character Linking for Multiparty Conversation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty (Eds.). Association for Computational Linguistics, Online, 539–548. https://doi.org/10.18653/v1/2021.eacl-main.43

[7] Krisztian Balog and Tom Kenter. 2019. Personal Knowledge Graphs: A Research Agenda. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval, ICTIR 2019, Santa Clara, CA, USA, October 2-5, 2019*, Yi Fang, Yi Zhang, James Allan, Krisztian Balog, Ben Carterette, and Jiafeng Guo (Eds.). ACM, New York, NY, USA, 217–220. https://doi.org/10.1145/3341981.3344241

[8] Rafael E. Banchs and Haizhou Li. 2012. IRIS: a Chat-oriented Dialogue System based on the Vector Space Model. In *The 50th Annual Meeting of the Association for Computational Linguistics, Proceedings of the System Demonstrations, July 10, 2012, Jeju Island, Korea*. The Association for Computer Linguistics, Jeju Island, Korea, 37–42. https://aclanthology.org/P12-3007/

[9] Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for Generative Multi-Hop Question Answering Tasks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 4220–4230. https://doi.org/10.18653/v1/d18-1454

[10] Adrian Benton and Mark Dredze. 2015. Entity Linking for Spoken Language. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar (Eds.). The Association for Computational Linguistics, Denver, Colorado, USA, 225–230. https://doi.org/10.3115/v1/n15-1024

[11] Kevin Bowden, JiaQi Wu, Shereen Oraby, Amita Misra, and Marilyn A. Walker. 2018. SlugNERDS: A Named Entity Recognition Tool for Open Domain Dialogue Systems. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*, Nicoletta Calzolari, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Kôiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (Eds.). European Language Resources Association (ELRA), Miyazaki, Japan. http://www.lrec-conf.org/proceedings/lrec2018/summaries/819.html

[12] Kevin Clark and Christopher D. Manning. 2016. Deep Reinforcement Learning for Mention-Ranking Coreference Models. *CoRR* abs/1609.08667 (2016). arXiv:1609.08667 http://arxiv.org/abs/1609.08667

[13] Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Mach. Learn.* 20, 3 (1995), 273–297. https://doi.org/10.1007/BF00994018

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[15] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). ACM, 855–864. https://doi.org/10.1145/2939672.2939754

[16] Ho Thao Hien, Pham-Nguyen Cuong, Le Nguyen Hoai Nam, Ho Le Thi Kim Nhung, and Thang Le Dinh. 2018. Intelligent Assistants in Higher-Education Environments: The FIT-EBot, a Chatbot for Administrative and Learning Support. In *Proceedings of the Ninth International Symposium on Information and Communication Technology, SoICT 2018, Danang City, Vietnam, December 06-07, 2018*. ACM, Danang City, Vietnam, 69–76. https://doi.org/10.1145/3287921.3287937

[17] Ben Hixon, Peter Clark, and Hannaneh Hajishirzi. 2015. Learning Knowledge Graphs for Question Answering through Conversational Dialog. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, Rada Mihalcea, Joyce Yue Chai, and Anoop Sarkar (Eds.). The Association for Computational Linguistics, Denver, Colorado, USA, 851–861. https://doi.org/10.3115/v1/n15-1086

[18] Filip Ilievski, Pedro A. Szekely, and Daniel Schwabe. 2020. Commonsense Knowledge in Wikidata. In *Proceedings of the 1st Wikidata Workshop (Wikidata 2020) co-located with 19th International Semantic Web Conference(OPub 2020), Virtual Conference, November 2-6, 2020 (CEUR Workshop Proceedings, Vol. 2773)*, Lucie-Aimée Kaffee, Oana Tifrea-Marciuska, Elena Simperl, and Denny Vrandecic (Eds.). CEUR-WS.org, Virtual. http://ceur-ws.org/Vol-2773/paper-10.pdf

[19] Filip Ilievski, Pedro A. Szekely, and Bin Zhang. 2021. CSKG: The CommonSense Knowledge Graph. In *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings (Lecture Notes in Computer Science, Vol. 12731)*, Ruben Verborgh, Katja Hose, Heiko Paulheim, Pierre-Antoine Champin, Maria Maleshkova, Óscar Corcho, Petar Ristoski, and Mehwish Alam (Eds.). Springer, Virtual, 680–696. https://doi.org/10.1007/978-3-030-77385-4_41

[20] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A Survey on Knowledge Graphs: Representation, Acquisition, and Applications. *IEEE Trans. Neural Networks Learn. Syst.* 33, 2 (2022), 494–514. https://doi.org/10.1109/TNNLS.2021.3070843

[21] Hideaki Joko, Faegheh Hasibi, Krisztian Balog, and Arjen P. de Vries. 2021. Conversational Entity Linking: Problem Definition and Datasets. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, Virtual Event, Canada, 2390–2397. https://doi.org/10.1145/3404835.3463258

[22] Heeyoung Lee, Angel X. Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. *Comput. Linguistics* 39, 4 (2013), 885–916. https://doi.org/10.1162/COLI_a_00152

[23] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. *CoRR* abs/1707.07045 (2017). arXiv:1707.07045 http://arxiv.org/abs/1707.07045

[24] Xiang Li, Gökhan Tür, Dilek Hakkani-Tür, and Qi Li. 2014. Personal knowledge graph population from user utterances in conversational understanding. In *2014 IEEE Spoken Language Technology Workshop, SLT 2014, South Lake Tahoe, NV, USA, December 7-10, 2014*. IEEE, South Lake Tahoe, NV, USA, 224–229. https://doi.org/10.1109/SLT.2014.7078578

[25] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-BERT: Enabling Language Representation with Knowledge Graph. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, New York, NY, USA,

2901–2908. https://ojs.aaai.org/index.php/AAAI/article/view/5681

[26] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692

[27] Alain Loisel, Guillaume Dubuisson Duplessis, Nathalie Chaignaud, Jean-Philippe Kotowicz, and Alexandre Pauchet. 2012. A Conversational Agent for Information Retrieval based on a Study of Human Dialogues. In *ICAART 2012 - Proceedings of the 4th International Conference on Agents and Artificial Intelligence, Volume 1 - Artificial Intelligence, Vilamoura, Algarve, Portugal, 6-8 February, 2012*, Joaquim Filipe and Ana L. N. Fred (Eds.). SciTePress, Algarve, Portugal, 312–317.

[28] Gideon S. Mann and David Yarowsky. 2003. Unsupervised Personal Name Disambiguation. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4* (Edmonton, Canada) *(CONLL '03)*. Association for Computational Linguistics, USA, 33–40. https://doi.org/10.3115/1119176.1119181

[29] Todor Mihaylov and Anette Frank. 2018. Knowledgeable Reader: Enhancing Cloze-Style Reading Comprehension with External Commonsense Knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, 821–832. https://doi.org/10.18653/v1/P18-1076

[30] Salman Mohammed, Peng Shi, and Jimmy Lin. 2018. Strong Baselines for Simple Question Answering over Knowledge Graphs with and without Neural Networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, Marilyn A. Walker, Heng Ji, and Amanda Stent (Eds.). Association for Computational Linguistics, New Orleans, Louisiana, USA, 291–296. https://doi.org/10.18653/v1/n18-2047

[31] Ryan Musa, Xiaoyan Wang, Achille Fokoue, Nicholas Mattei, Maria Chang, Pavan Kapanipathi, Bassem Makni, Kartik Talamadupula, and Michael Witbrock. 2019. Answering Science Exam Questions Using Query Reformulation with Background Knowledge. In *1st Conference on Automated Knowledge Base Construction, AKBC 2019, Amherst, MA, USA, May 20-22, 2019*. https://doi.org/10.24432/C5CC7R

[32] Janna Omeliyanenko, Albin Zehe, Lena Hettinger, and Andreas Hotho. 2020. LM4KG: Improving Common Sense Knowledge Graphs with Language Models. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 12506)*, Jeff Z. Pan, Valentina A. M. Tamma, Claudia d'Amato, Krzysztof Janowicz, Bo Fu, Axel Polleres, Oshani Seneviratne, and Lalana Kagal (Eds.). Springer, Athens, Greece, 456–473. https://doi.org/10.1007/978-3-030-62419-4_26

[33] Matthew E. Peters, Mark Neumann, Robert L. Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, 43–54. https://doi.org/10.18653/v1/D19-1005

[34] Hongjin Qian, Xiaohe Li, Hanxun Zhong, Yu Guo, Yueyuan Ma, Yutao Zhu, Zhanliang Liu, Zhicheng Dou, and Ji-Rong Wen. 2021. Pchatbot: A Large-Scale Dataset for Personalized Chatbot. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, Canada, 2470–2477. https://doi.org/10.1145/3404835.3463239

[35] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained Models for Natural Language Processing: A Survey. *CoRR* abs/2003.08271 (2020). arXiv:2003.08271 https://arxiv.org/abs/2003.08271

[36] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: An Atlas of Machine Commonsense for If-Then Reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, Honolulu, Hawaii, USA, 3027–3035. https://doi.org/10.1609/aaai.v33i01.33013027

[37] Jürgen Schmidhuber. 2014. Deep Learning in Neural Networks: An Overview. *CoRR* abs/1404.7828 (2014). arXiv:1404.7828 http://arxiv.org/abs/1404.7828

[38] Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, Berlin, Germany. https://doi.org/10.18653/v1/p16-1162

[39] Mingyue Shang, Tong Wang, Mihail Eric, Jiangning Chen, Jiyang Wang, Matthew Welch, Tiantong Deng, Akshay Grewal, Han Wang, Yue Liu, Yang Liu, and Dilek Hakkani-Tur. 2021. Entity Resolution in Open-domain Conversations. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*. Association for Computational Linguistics, Online, 26–33. https://doi.org/10.18653/v1/2021.naacl-industry.4

[40] SpaCY. 2021. Spacy Documentation Model Architectures. https://spacy.io/api/architectures#EntityLinker.

[41] Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, Satinder P. Singh and Shaul Markovitch (Eds.). AAAI Press, San Francisco, California, USA, 4444–4451. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972

[42] Dianbo Sui, Yubo Chen, Kang Liu, Jun Zhao, Xiangrong Zeng, and Shengping Liu. 2020. Joint Entity and Relation Extraction with Set Prediction Networks. *CoRR* abs/2011.01675 (2020). arXiv:2011.01675 https://arxiv.org/abs/2011.01675

[43] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. 2020. CoLAKE: Contextualized Language and Knowledge Embedding. In *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, Donia Scott, Núria Bel, and Chengqing Zong (Eds.). International Committee on Computational Linguistics, Online, 3660–3670. https://doi.org/10.18653/v1/2020.coling-main.327

[44] N. T. Thomas. 2016. An e-business chatbot using AIML and LSA. In *2016 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2016, Jaipur, India, September 21-24, 2016*. IEEE, Jaipur, India, 2740–2742. https://doi.org/10.1109/ICACCI.2016.7732476

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (Eds.). NeurIPS, Long Beach, CA, USA, 5998–6008. https://proceedings.neurips.cc/paper/2017/hash/

3f5ee243547dee91fbd053c1c4a845aa-Abstract.html

1205

[46] Anu Venkatesh, Chandra Khatri, Ashwin Ram, Fenfei Guo, Raefer Gabriel, Ashish Nagar, Rohit Prasad, Ming Cheng, Behnam Hedayatnia, Angeliki Metallinou, Rahul Goel, Shaohua Yang, and Anirudh Raju. 2018. On Evaluating and Comparing Conversational Agents. *CoRR* abs/1801.03625 (2018). arXiv:1801.03625 http://arxiv.org/abs/1801.03625

[47] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics* 9 (03 2021), 176–194. https://doi.org/10.1162/tacl_a_00360 arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00360/1923927/tacl_a_00360.pdf

[48] Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue Natural Language Inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, Anna Korhonen, David R. Traum, and Lluís Màrquez (Eds.). Association for Computational Linguistics, Florence, Italy, 3731–3741. https://doi.org/10.18653/v1/p19-1363

[49] Chien-Sheng Wu, Andrea Madotto, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2020. Getting To Know You: User Attribute Extraction from Dialogues. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, Nicoletta Calzolari, Frédéric Béchet, Philippe Blache, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, and Stelios Piperidis (Eds.). European Language Resources Association, Marseille, France, 581–589. https://aclanthology.org/2020.lrec-1.73/

[50] Mengge Xue, Weiming Cai, Jinsong Su, Linfeng Song, Yubin Ge, Yubao Liu, and Bin Wang. 2019. Neural Collective Entity Linking Based on Recurrent Random Walk Network Learning. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, Sarit Kraus (Ed.). ijcai.org, 5327–5333. https://doi.org/10.24963/ijcai.2019/740

[51] Xiaoxin Yin, Jiawei Han, and Philip S. Yu. 2007. Object Distinction: Distinguishing Objects with Identical Names. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, Rada Chirkova, Asuman Dogac, M. Tamer Özsu, and Timos K. Sellis (Eds.). IEEE Computer Society, The Marmara Hotel, Istanbul, Turkey, 1242–1246. https://doi.org/10.1109/ICDE.2007.368983

[52] SoYeop Yoo and OkRan Jeong. 2020. Automating the expansion of a knowledge graph. *Expert Syst. Appl.* 141 (2020). https://doi.org/10.1016/j.eswa.2019.112965

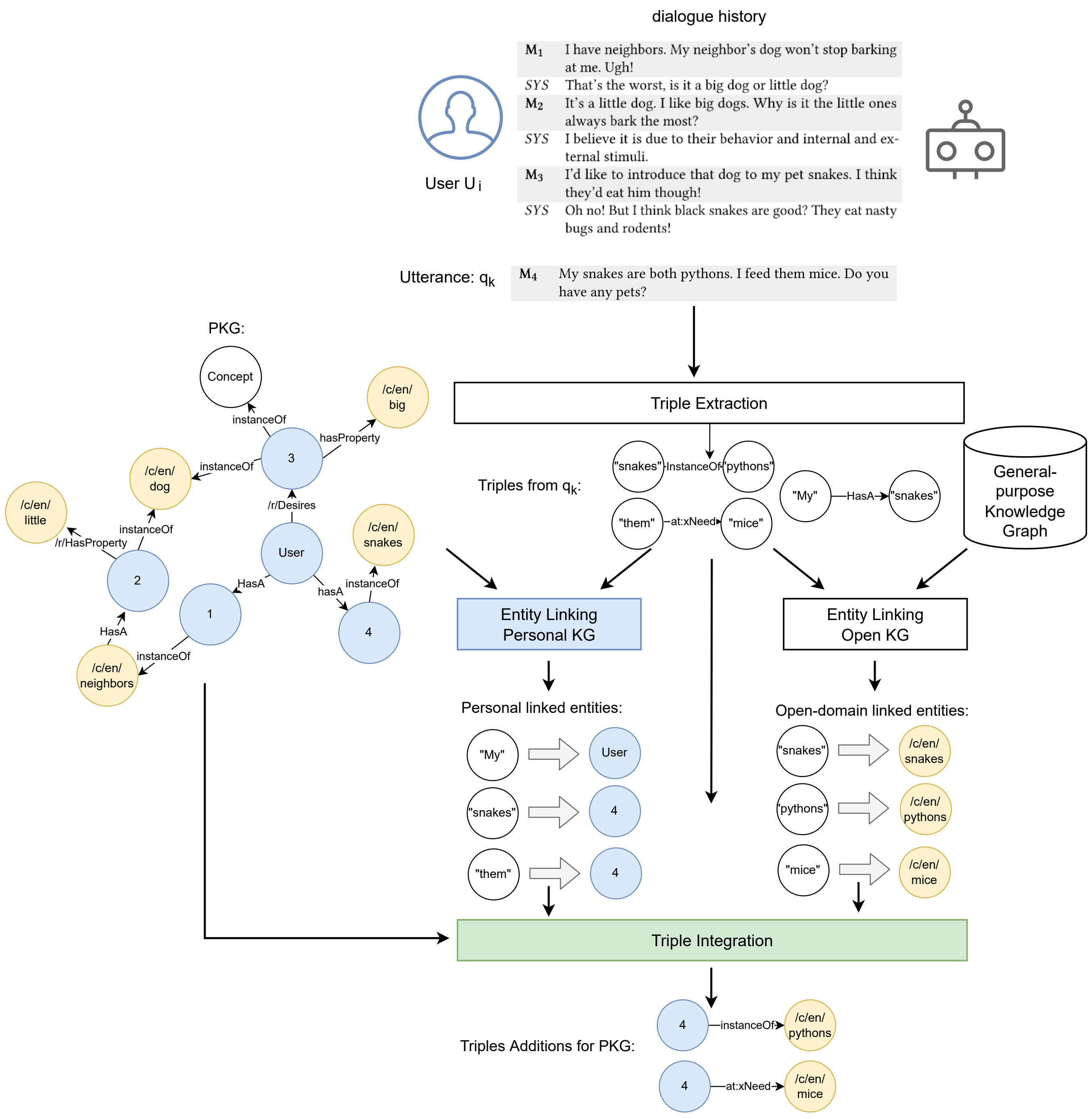
[53] Munazza Zaib, Quan Z. Sheng, and Wei Emma Zhang. 2020. A Short Survey of Pre-trained Language Models for Conversational AI-A New Age in NLP. In *Proceedings of the Australasian Computer Science Week, ACSW 2020, Melbourne, VIC, Australia, February 3-7, 2020*, Prem Prakash Jayaraman, Dimitrios Georgakopoulos, Timos K. Sellis, and Abdur Forkan (Eds.). ACM, Melbourne, VIC, Australia, 11:1–11:4. https://doi.org/10.1145/3373017.3373028

[54] Hongming Zhang, Xinran Zhao, and Yangqiu Song. 2020. A Brief Survey and Comparative Study of Recent Development of Pronoun Coreference Resolution. *CoRR* abs/2009.12721 (2020). arXiv:2009.12721 https://arxiv.org/abs/2009.12721

[55] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing Dialogue Agents: I have a dog, do you have pets too?. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, Melbourne, Australia, 2204–2213. https://doi.org/10.18653/v1/P18-

A DATA FLOW

To further explain the architecture, we have elaborated it with a chart of the data flow. This data flow chart is displayed on **Figure 10** and displays the passed data through nodes to highlight the graph structure.



M_1	I have neighbors. My neighbor's dog won't stop barking at me. Ugh!
SYS	That's the worst, is it a big dog or little dog?
M_2	It's a little dog. I like big dogs. Why is it the little ones always bark the most?
SYS	I believe it is due to their behavior and internal and external stimuli.
M_3	I'd like to introduce that dog to my pet snakes. I think they'd eat him though!
SYS	Oh no! But I think black snakes are good? They eat nasty bugs and rodents!

M_4	My snakes are both pythons. I feed them mice. Do you have any pets?

Figure 10. The data flow chart of the improved architecture instantiated with a specific conversation data. This figure shows a graphical representation of each step. Our problem formulations are encapsulated by the *Entity Linking Person KG* and *Triple Integration* components.

B PATTERNS FOR CONVERSATION RANKING

Table 7. Patterns for Ranking of Conversations. The relations in this table are presented in **Table 8**

Patterns
Possessive pronouns[5] are included in at least 1 utterance in the conversation
The pattern "*personal pronoun*[6] * *ownership_relation* *" is matched by an utterance
The pattern "*personal pronoun* * *preference_relation* *" is matched by an utterance
The pattern "*personal pronoun* * *aversion_relation* *" is matched by an utterance

Table 8. alternatives of relations used in Patterns

Relation	Synonyms
owner_relation	has, have, own(s), posses(s), keep(s), retain(s), am, are
preference_relation	like(s), prefer(s), love(s), desire(s), fancy, fancies, enjoy(s), appreciate(s), admire(s), cherish(es)
aversion_preference	hate(s) loath(s) despise(s) dislike(s), resent(s), detest(s), disapprove(s), deprecate(s)

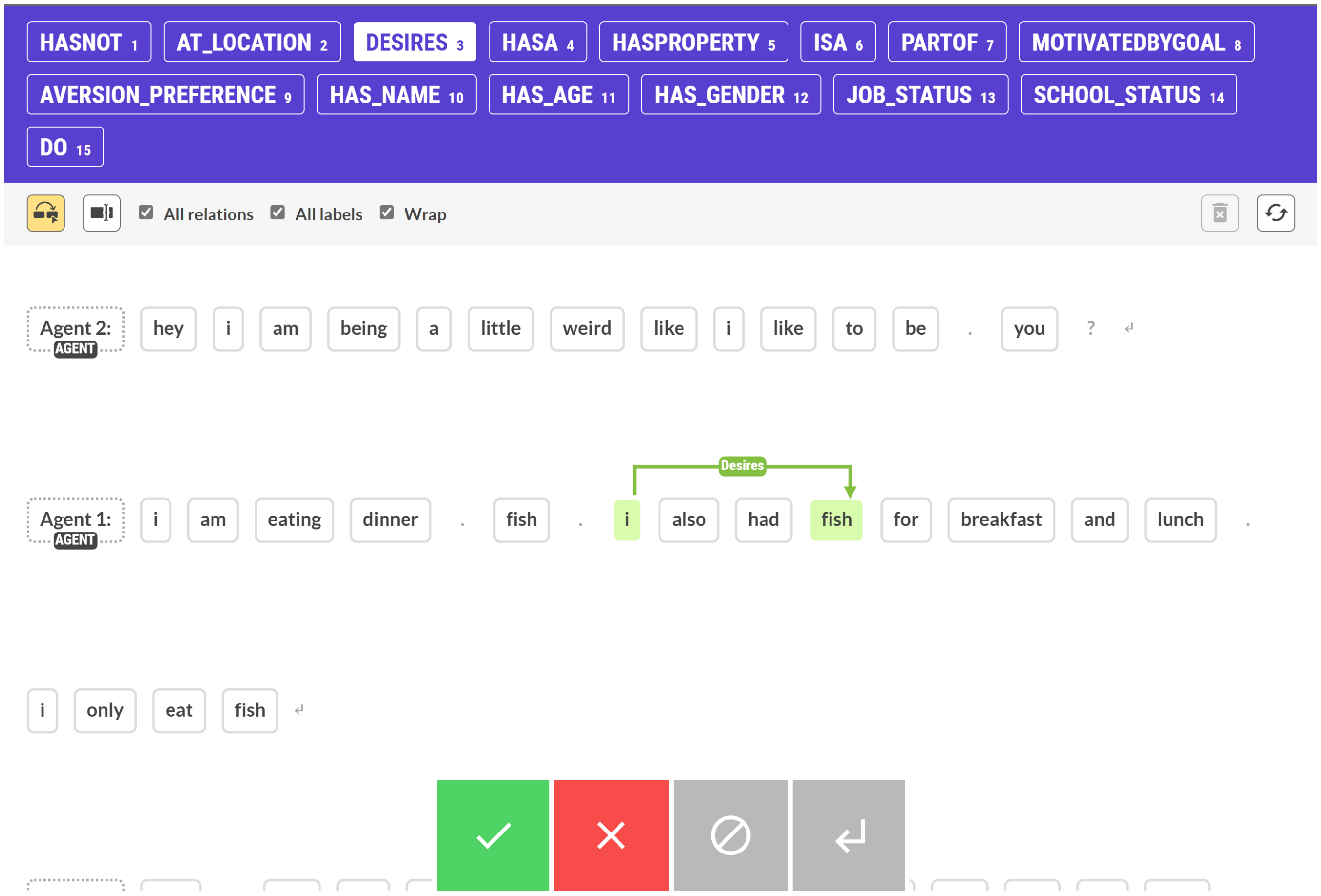
[5]Possesive pronouns are "my", "your", "his", "her", "its", "our", and "their"
[6]Personal pronouns refers to "I" and "We"

C ANNOTATION WEBSITES

To create our dataset, we implemented three different annotation interfaces to gather textual triple annotations, open KG annotations and personal entity annotations. We present a sample of the websites.

C.1 Triple Annotation Website

To gather the textual triple annotations, we created the website displayed in Figure 11. The top purple bar displays the different options of relations to select when annotating. The user will then have to select one of these relations and then select the two entity mentions from the text that has this relation. As illustrated on the figure, we have selected the *Desires* and then chosen the entity mention "*I*" as subject and entity mention "*fish*" as object. This triple is then highlighted in green. After annotating the entire conversation, the user can then click the green check mark to approve the conversation with annotations.



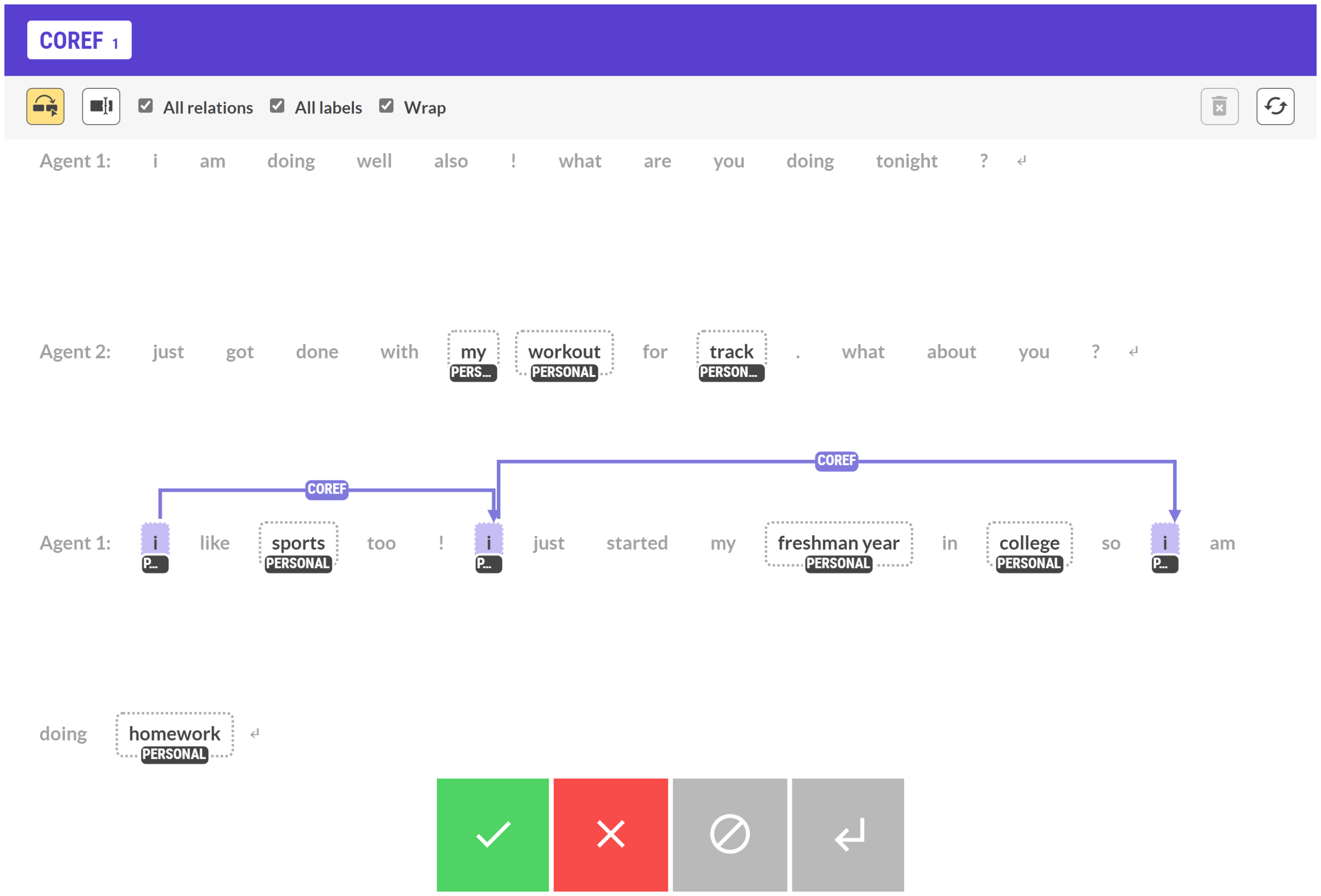
Agent 2: hey i am being a little weird like i like to be . you ?

Agent 1: i am eating dinner . fish . i also had fish for breakfast and lunch . i only eat fish

Figure 11. Illustation of the URG. The blue nodes are word nodes. The green nodes are relation nodes, and the yellow nodes are ConceptNet entity nodes.

C.2 Personal Entity Annotation Website

To gather the personal entity annotations, we created the website displayed in **Figure 12**. This website is quite similar to the aforementioned website for triple annotations in **Figure 11**. The only difference is that there is only a single relation now, namely *COREF*. This relation denotes the relation between entity mentions that reference the same real world entity. They are annotated as a chain as seen on the figure, where all textual mentions of "*I*" are linked, since these all refer to the user. After annotating the entire conversation, the user can once again click the green check mark to approve the conversation with annotations.



Agent 1: i am doing well also ! what are you doing tonight ?

Agent 2: just got done with my workout for track . what about you ?

Agent 1: i like sports too ! i just started my freshman year in college so i am doing homework

Figure 12. Illustation of the URG. The blue nodes are word nodes. The green nodes are relation nodes, and the yellow nodes are ConceptNet entity nodes.

D SAMPLE OF ANNOTATION DATA

D.1 Textual Triple Annotation

Listing 1. Structure of textual triple annotations in JSONL. The sample is of one line in the data file

```

{
    "text": "Agent 1: do you play any sports ? i run
         track\n Agent 2: i like to walk but that is
         about it . i am a busy mother ...",
    "conv_id": 19,
    "relations": [
                 {
                   "head_span": {
                       "text": "i",
                       "start": 34,
                       "end": 35,
                       "label": "ENTITY"
                  },
                   "child_span": {
                       "text": "track",
                       "start": 40,
                       "end": 45,
                       "label": "ENTITY"
                   },
                   "label": "Desires"
                   }, ...
                 ],
}
```

D.2 ConceptNet Annotation Sample

Listing 2. Structure of personal entity annotations in JSONL. The sample is of one line in the data file

```

{
    "text": " Agent 2: just got done with my workout
         for track . what about you ?",
    "conv_id": 0,
    "spans":[{
        "text":"workout",
        "start":32,
        "end":39,
        "label":"ENTITY"}
        ],"turn":3,
        "options":[{"id":"/c/en/workout"},{"id":"/c/en
             /workouts"},{"id":"/c/en/workout/n"},{"id
             ":"/c/en/workouts/n"},{"id":"/c/en/
             preworkout"}],
        "accept":["/c/en/workout"]}
```

D.3 Personal Entity Annotations

Listing 3. Structure of ConceptNet entity annotations in JSONL. The sample is of one line in the data file

```
{
    "conv_id": 4,
    "text": "Agent 1: hi how is your day going so far\
         n Agent 2: i am doing good ...",
    "spans": [{
            "text": "Agent 2:",
            "start": 42,
            "token_start": 12,
            "token_end": 14,
            "end": 50,
            "type": "pattern",
            "label": "AGENT"
        },
        {
            "text": "pancakes",
            "start": 111,
            "token_start": 29,
            "token_end": 29,
            "end": 119,
            "type": "span",
            "label": "PERSONAL"
        },
        {
            "text": "syrup",
            "start": 124,
            "token_start": 31,
            "token_end": 31,
            "end": 129,
            "type": "span",
            "label": "PERSONAL"
        },],
    "relations": [{
            "head": 37,
            "child": 103,
            "head_span": {
                "start": 145,
                "end": 146,
                "token_start": 37,
                "token_end": 37,
                "label": "PERSONAL"
            },
            "child_span": {
                "start": 409,
                "end": 410,
                "token_start": 103,
                "token_end": 103,
                "label": "PERSONAL"
            },
            "label": "COREF"
        }, ...
```

E SAMPLE OF ANNOTATION DATA POST PROCESSED

Listing 4. Structure of conversational data after post processing the annotations. The sample is of one line in the data file

```
{
    "conv_id": 0,
    "utterances": [
                   {
                       "text": "i like sports too ! i just started my freshman year in college so i am doing
                            homework",
                       "relations": [
                           {
                               "head_span": {
                                   "text": "i",
                                   "start": 0,
                                   "end": 1,
                                   "label": "ENTITY",
                                   "personal_id": 0
                               },
                               "child_span": {
                                   "text": "sports",
                                   "start": 7,
                                   "end": 13,
                                   "label": "ENTITY",
                                   "conceptnet": "/c/en/sports",
                                   "personal_id": 1
                               },
                               "label": "Desires"
                           },...
                       ]
            ]
}
```

F EXAMPLE OF EMBEDDING LAYER

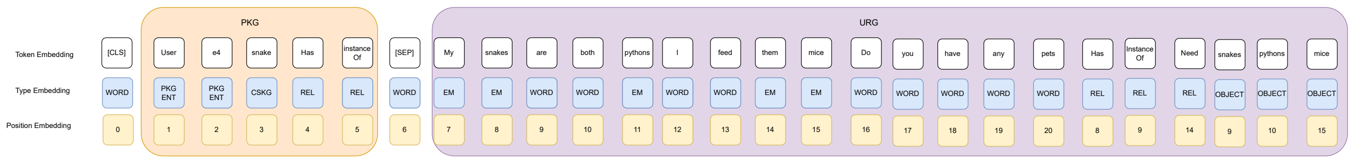

PKG
URG
Token Embedding
Type Embedding
Position Embedding
[CLS] User e4 snake Has instance Of [SEP] My snakes are both pythons I feed them mice Do you have any pets Has Instance Of Need snakes pythons mice
WORD PKG ENT PKG ENT CSKG REL REL WORD EM EM WORD WORD EM WORD WORD EM EM WORD WORD WORD WORD WORD REL REL REL OBJECT OBJECT OBJECT
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 8 9 14 9 10 15

Figure 13. Example which illustrates the intuition behind the embedding layer to our PEC.

G URG CONSTRUCTION WITH CSKG ENTITIES

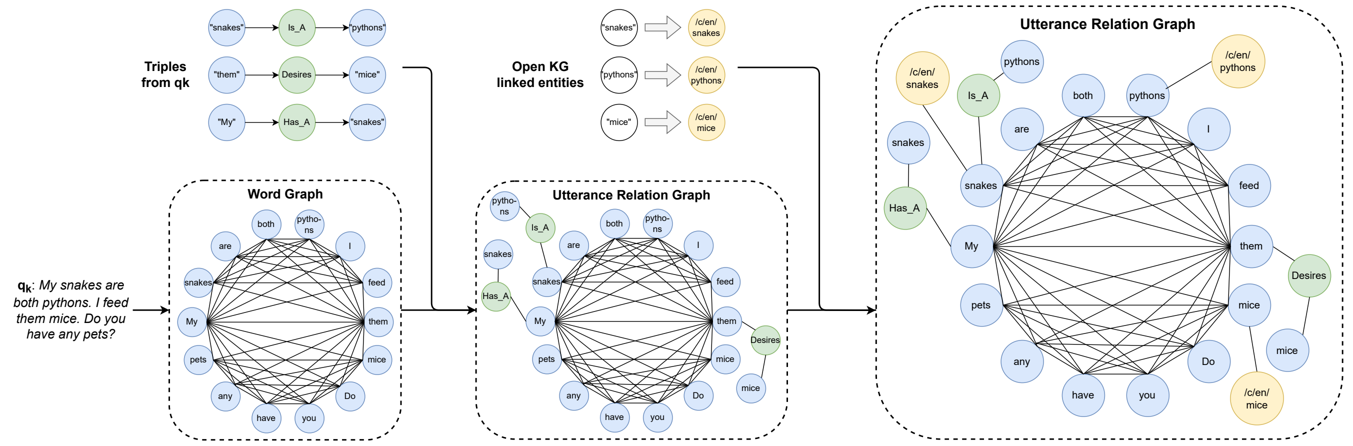


Figure 14. Illustation of the URG. The blue nodes are word nodes. The green nodes are relation nodes, and the yellow nodes are ConceptNet entity nodes.

H ARCHITECTURE

To elaborate the architecture with additional components to reflect the PKG Statement Linking problem and PKG Enrichment problem. This extended architecture is displayed in **Figure 15**. From this figure, it is clear to see that we have added the *Entity Linking Personal KG* component, which reflects the PKG Statement Linking problem. The PKG Enrichment problem is reflected in the component denoted as *Triple Integration*.

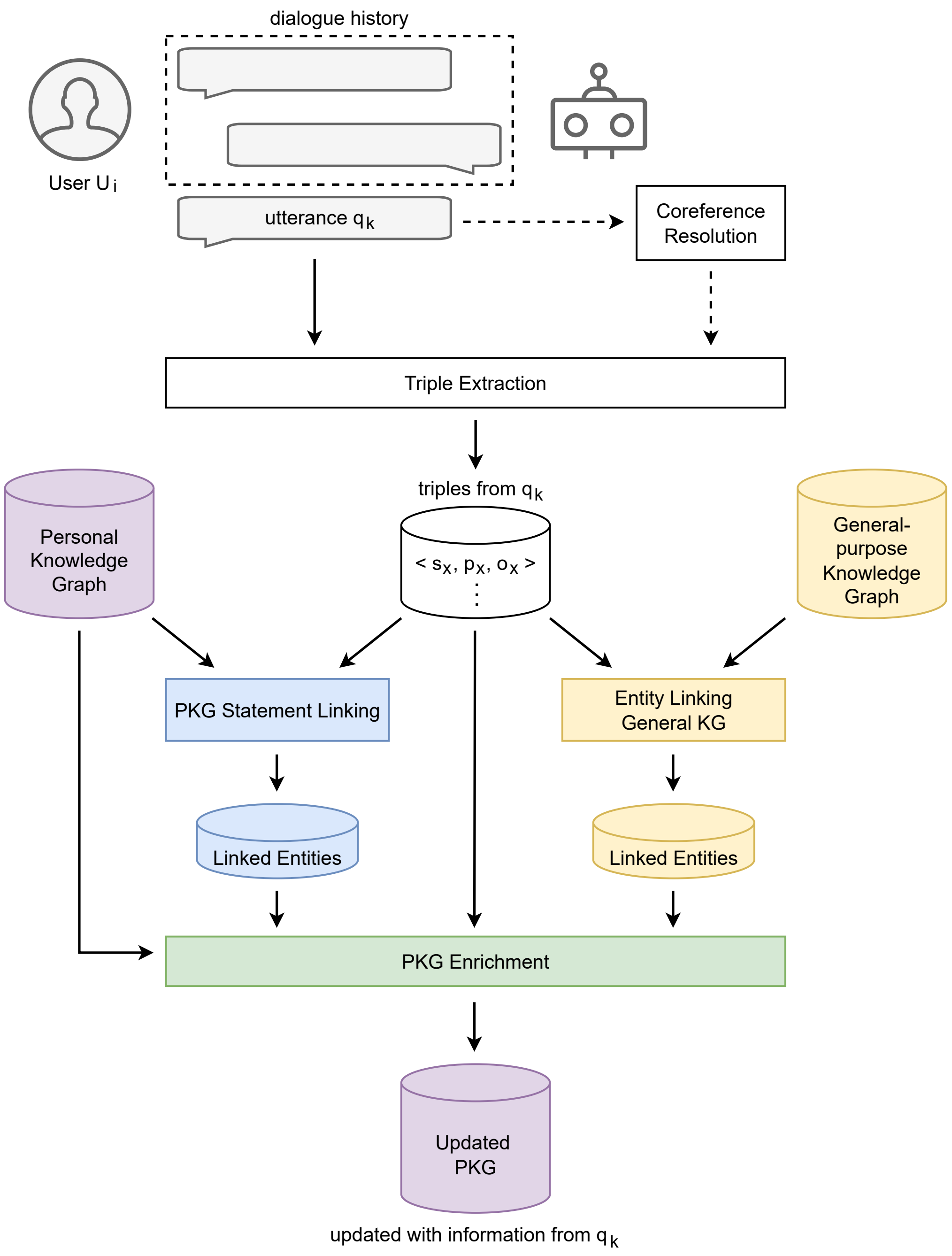


Figure 15. The architecture extended to capture the input/output data for the different phases. In blue, we observe the entity linking to a PKG for user U_i. The PKG is built using the dialogue history. The last element in the architecture is the updated PKG now containing personal information capture from utterance q_k also.

I CONSIDERATIONS FOR DATASET CREATION

We present some of our initial thoughts regarding the data collection and problem definitions.

I.1 Thoughts on input to problem definition

We could include the textual triples as input instead of just the recognised entity mentions as this would enable the option to traverse the PKG based on these relations to better disambiguate the personal entities in the PKG, e.g., if we have PKG with two dog entities but they are connected to the user differently, the relations information could help identify which of the entities is the correct one. This section centers on dataset creation to highlight important information from other papers on creating personalised datasets. The ideal dataset for PKG population from conversations is conversational data, where the user utterances are annotated with ground truth triples and entities.

I.2 Conversational Data

Acquiring a conversational dataset can be quite difficult and expecially a large-scale one. Qian et al. [34] present their large-scale, chinese, conversational dataset named Pchatbot. They have created this by combining several exisiting conversational datasets and applying some filtering to protect user privacy and relevance of data. Therefore it might be unnecessary for us to collect the conversational data ourselves. Since we can utilise datasets as analysed in the prior semester, such as PersonaChat. Though we still need triple annotations and linked entity annotations for the dataset to be ideal for our use case.

For the initial dataset, we are only going to use the PersonaChat dataset. This is due to the high level of personal information provided in these conversations, which is necessary to populate a PKG.

I.3 Triple Annotations

Last semester, we worked with the *Textual Triple Annotated Dataset* presented by [49], which was already annotated with triples. These triples were not ideal for Problem Definition 2 presented in our previous work, since they rarely had triples that would be include information more than 1-hop away from the user. We therefore examine different ways to gathering triple annotations.

We could use SPN4RE presented by Sui et al. [42] to create the triple annotations. Though, we did observe last semester that these were not completely ideal and still contained quite a lot of annotation errors. Similarly, the triple annotator OpenIE presented in Angeli et al. [4] performed quite poorly to our expectations. Another possibility would be to crowd source triple annotations. This approach was used the triple annotations on the DialogueNLI dataset[48]. This was done by themselves annotating the persona sentences of the PersonaChat dataset and then providing a limited set of entities and relations to the crowd source workers, which they could triple annotate messages with. An alternative to perform triple annotations could be to use the triple annotators SPN4RE and OpenIE to create a set of possible triple annotations. These can then be presented to a crowd source worker to evaluate, which triples are right depending on the utterance and whether additional triples are required.

From the previously mentioned approaches to create triple annotations, we have decided to annotate a subset of the dataset with triples ourselves, such that we can train SPN4RE on this data and use this model to create the triple annotation on the utterances of the conversation.

I.4 Entity Annotations

We also need ground truth linked entity annotations to dataset to ensure that the right information is captured by the entity linker. To this end, we need to differentiate between two types of entities, namely general-purpose entities and personal entities since these can be linked through two different knowledge graphs.

I.4.1 General-Purpose Knowledge Graph. A possible approach to gather the entity annotations would be to use these entity linking tools to create a set of possible linked entities. To get additional possible linked entities could be to search the entity mentions on the general-purpose knowledge graphs web API. Then we can select top-k search results as potential entities. We can then prompt a user to select between the set of possible entities to only annotate with the correct linked entities. An alternative approach to create linked entity annotations would be to adopt a similar approach to the triple annotations, where we could use an annotation tool like Prodigy to annotate a smaller dataset ourselves and then use spacy to train an entity linker, which we can then use of the rest of the data. Thereby, creating entity annotations in a distantly supervised fashion.

We have chosen that the most optimal approach is to use a general-purpose knowledge graph with a web API, since this would make it quite easier to create a set of possible linked entities from the top-k elements in the search results of an entity mention.

I.4.2 Personal Knowledge Graph. Gathering personal entities might be a bit more of a difficult task, since this is quite limited with research. We observe that this task has resemblance to the task of coreference resolution.

We have chosen to gather the personal entity annotations ourselves by annotating a set of conversations with coreference chains.

I.5 Annotation on conversations

We describe what the annotations should be for the problem formulations. Essentially with these annotations, it should be possible to construct a golden standard PKG for specific users by replacing the entity mentions in the triples with the corresponding personal entity and adding a triple for each overlap between General KG entity and personal entity with the type relation.

How to model relation between linked entities to general kg to personal entities?

Check when creating dataset whether literals should be included.

I.6 Dataset Creation Process

We have decided to create a website for annotating data. To understand this system in more depth, we have created prototypes of how we expect to annotate this data.

Figure 5 displays a low-level prototype of what the entity annotation described in subsection I.4. It simply displays an utterance and top-5 search results given the entity mention. We highlight the entity mention so it is easy to identify from it's context. We use radio-buttons since only a single entity is needed for linking. The user can select the most appropriate entity and 'submit'.

For the triple annotations, we expect to annotate a set of data ourselves and then train a model on this data to annotate the rest. We can then prompt the user for whether these annotations are correct.

Prodigy To create the data annotation website, we have decided to utilise the tool, Prodigy. Prodigy is an annotation tool which makes it easy to highlight entities in sentences and mark relationships between entities.

I.7 Evaluation of Problem Formulation Using Dataset

We have two possible settings for evaluating the constructed PKG; one on a utterance level, and another on the conversation level.

On the utterance level, we can construct a PKG based on the ground truth annotations from the previous utterances and use this as the input PKG for specific utterances. This has the benefit that errors will not be over-represented, e.g., if a method models the PKG incorrectly in the previous step, it is possible that information is missing to correctly link to the PKG, which makes this not the method's fault.

On the conversational level, the previously constructed PKG based on the method itself on the previous conversation can be used as the input PKG. This is more aligned with a real world scenario.

I.8 Samples of Data Annotations

Listing 5. Sample of annotated conversation

```
{
partner's persona: i like canning and whittling. ['i',
     'like_activity', 'whittling']
partner's persona: to stay in shape , i chase
    cheetahs at the zoo. ['i', 'like_activity', '
    chasing cheetahs']
partner's persona: in high school , i came in 6th in
    the 100 meter dash. ['i', 'has_ability', '6 100
    meter dash']
partner's persona: i eat exclusively meat. ['i', '
    other', 'carnivore']
your persona: i like to remodel homes. ['i', '
    like_general', 'remodelling homes']
your persona: i like to go hunting. ['i', '
    like_activity', 'hunting']
your persona: i like to shoot a bow. ['i', '
    like_sports', 'archery']
your persona: my favorite holiday is halloween. ['i',
     'favorite', 'halloween']
1 hi , how are you doing ? i am getting ready to do
    some cheetah chasing to stay in shape . { "
    triples": [[25:26], 'like_activity', [55:70]], "
    linked_entities": [([55-62], KG:Cheetah)], "
    personal_entities":[([25-26],0)]}
2 you must be very fast . hunting is one of my
    favorite hobbies . ['i', 'like_activity', '
    hunting']
3 i am ! for my hobby i like to do canning or some
    whittling . ['i', 'like_activity', 'whittling']
4 i also remodel homes when i am not out bow hunting .
     ['i', 'like_activity', 'hunting'] ['i', '
    like_general', 'remodelling homes']
5 that is neat . when i was in high school i placed 6
    th in 100m dash ! ['i', 'has_ability', '6 100
    meter dash']
6 that is awesome . do you have a favorite season or
    time of year ?
7 i do not . but i do have a favorite meat since that
     is all i eat exclusively . ['i', 'other', '
    carnivore']
8 what is your favorite meat to eat ?
9 i would have to say its prime rib . do you have any
     favorite foods ?
10 i like chicken or macaroni and cheese .
11 do you have anything planned for today ? i think i
     am going to do some canning . ['i', '
    like_activity', 'whittling']
12 i am going to watch football . what are you
    canning ?
13 i think i will can some jam . do you also play
    footfall for fun ?
```

Listing 6. Structure of data in JSON

```
"dialogues": [
    {
        "id": 7,
        "utterances": [{
                "utterance": "my mom had me in mcdonald
                      bathroom when she was 12",
                "turn": 1,
                "textual triples": [
                    [[0:2], 'born_in', [17:25]],
                    [[0:2], 'have', [3:6]],
                    [[3:6], 'pregnant_in', [48-50]]
                ],
                "linked_entities to general KG": [
                    ([3:6], "Q7560"),
                    ([17:25], Q_MCDONNALD),
                    ([26:35], Q190771)
                ],
                "personal entities": [
                    ([0:2], 0) #user node for one agent
                    ([3:6], 1) # mother node substring:
                          "mom"
                    ([40:43], 1) # mother node
                         substring: "she"
                ]
                },
                { ...
```