Extracting and Inferring Personal Attributes from Dialogue

Zhilin Wang, Xuhui Zhou, Rik Koncel-Kedziorski, Alex Marin, Fei Xia
University of Washington
{zhilinw, xuhuizh, kedzior, amarin, fxia}@uw.edu

Abstract

Personal attributes represent structured information about a person, such as their hobbies, pets, family, likes and dislikes. In this work, we introduce the tasks of extracting and inferring personal attributes from human-human dialogue. We first demonstrate the benefit of incorporating personal attributes in a social chit-chat dialogue model. Thus motivated, we propose the tasks of personal attribute extraction and inference, and then analyze the linguistic demands of these tasks. Based on this analysis, we introduce a model that combines constrained attribute generation in GPT2 with a BERT reranker to meet the specific challenges of these tasks. Not only is our model competitive on personal attribute extraction, our model outperforms strong baselines on inferring personal attributes that are not contained verbatim in the original text due to commonsense/lexical inferences, which occur frequently in everyday conversation.

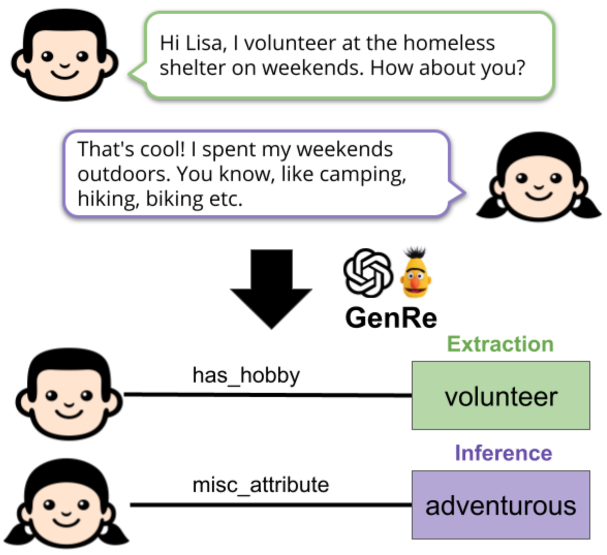
Hi Lisa, I volunteer at the homeless shelter on weekends. How about you?
That's cool! I spent my weekends outdoors. You know, like camping, hiking, biking etc.
GenRe
Extraction
has_hobby
volunteer
Inference
misc_attribute
adventurous

Figure 1: Overview of obtaining personal attribute triple from utterances using our model GenRe. Attribute values are contained within the utterance in the EXTRACTION task, but not the INFERENCE task.

1 Introduction

Personal attributes are structured information about a person such as what they like, what they have, and what their favorite things are. These attributes are commonly revealed either explicitly or implicitly during social dialogue as shown in Figure 1, allowing people to know more about one another.

A promising line of dialogue research focuses on representing users with relational knowledge graph (KG) triples (Li et al., 2014; Qian et al., 2018; Zheng et al., 2020a,b). KGs can represent large numbers of personal attributes in an interpretable manner (Hogan et al., 2021), and can facilitate reasoning for downstream tasks such as personalization of chit-chat dialogue (Majumder et al., 2020) or personalized recommendations in task-oriented dialogue (Pei et al., 2021). However, techniques for extracting personal attributes from dialogue are hampered by limited training data. Human utterances often contain multiple different types of personal information that can be annotated. Existing datasets (Li et al., 2016; Yu et al., 2020) tend to focus on demographic information and familial relationships. Those that do focus on personal attributes are distantly supervised using heuristics (Tigunova et al., 2019, 2020; Mazaré et al., 2018; Wu et al., 2019).

In Section 3, we show how personal attributes in the form of KG triples can improve the personalization of open-domain social chit-chat agents. We augment BlenderBot (Roller et al., 2020), a state-of-the-art generative model on the PersonaChat task (Zhang et al., 2018) with personal attributes to further improve its performance. Thus motivated, we introduce two new tasks for identifying personal attributes in Section 4. As shown in Figure 1, the EXTRACTION task requires determining which keyphrase in an utterance indicate a personal attribute, while the INFERENCE task adds further challenge by requiring models to predict personal attributes that are not explicitly stated verbatim in utterances. This is common in conversational settings, where people express personal attributes using a variety of semantically related words or imply them using commonsense reasoning. We analyze how these factors allow personal attributes to be linked to utterances that express them.

In Section 6, our experiments suggest that, while state-of-the-art Relation Extraction models are effective on the EXTRACTION task, they have ample space for improvement on the INFERENCE task. To address this shortcoming in Section 5, we propose a new model **GenRe** which takes advantage of pre-trained language models to generate and rank potential attribute candidates. Our model combines the flexibility of a GPT2-based constrained attribute generation model with a powerful BERT-based reranker. Finally in Section 7, detailed ablation studies demonstrate the value of these components to our tasks while further analysis identifies future areas for improvement.

2 Task Background

PersonaChat (Zhang et al., 2018) was created to improve the personality consistency of open-domain chit-chat dialogue agents. PersonaChat was constructed by giving two crowdworkers each a set of English personal attribute related sentences and asking them to chat in a way that is congruent with those sentences. Models were then trained to generate dialogue responses that are in line with those expressed by crowdworkers using the provided persona information as context.

DialogNLI (Welleck et al., 2019) contains samples of PersonaChat utterances, each paired with a manually annotated personal attribute triple. Each triple consists of a head entity, a relation, and a tail entity. These triples were initially annotated to identify entailing, contradicting and neutral statements within the PersonaChat corpus. For instance, a statement labelled with (I, [has_profession], chef) will contradict with another statement labelled with (I, [has_profession], engineer).

By re-purposing the DialogNLI dataset, our task seeks to extract these personal attribute triples from their paired utterances. The three largest groups of relations are: a. *has_X* (where X = *hobby, vehicle, pet*) b. *favourite_Y* (where Y = *activity, color, music*) c. *like_Z* (where Z = *read, drink, movie*).

3 Personal Attributes in Practice

Personal attributes can potentially make social chit-chat agents be more consistent and engaging as well as enable task-oriented agents to make personalized recommendations, as detailed in Section 8. As an example, we empirically demonstrate the value of personal attribute triples in personalizing open-domain social chit-chat by applying personal attributes to the PersonaChat task. We first explain how we utilize personal attributes in PersonaChat, then explain the evaluation metrics, and finally discuss our findings.

3.1 Methods

We fine-tune the generative version of Blender 90M (a transformer-based model trained on multiple related tasks) on PersonaChat, which is currently the state-of-the-art generative model on this task. We prepend a corresponding DialogNLI personal attribute before each utterance (*i.e.* **+Per. Attr.**), in order to better direct the model in generating a suitable response. This modification is relatively minimal to demonstrate the informativeness of personal attributes, while keeping the model architecture and hyperparameter fine-tuning the same as in the original work (details in Appendix A.1).

3.2 Metrics

Our metrics follow Roller et al. (2020) and Dinan et al. (2019). Metrics for **+Per. Attr.** setting consider both personal attributes and utterances. **Hits@1** uses the hidden states of the generated output to select the most likely utterance amongst 20 candidates (the correct utterance and 19 randomly chosen utterances from the corpus). **Perplexity** reflects the quality of the trained language model. **F1** demonstrates the extent of the overlap between the generated sequence and the ground truth sequence.

3.3 Results

	Hits@1 ↑	Perplexity ↓	F1 ↑
Blender	32.3 (σ=0.055)	11.3 (σ=0.0075)	20.4 (σ=0.091)
+ Per. Attr.	35.2 (σ=0.077)	10.4 (σ=0.0083)	20.6 (σ=0.14)

Table 1: Effects of using personal attributes to augment Blender on Personachat. Standard deviation is based on 5 separate runs. Higher is better for Hits@1 and F1; lower is better for perplexity.

As shown in Table 1, including information from personal attributes can improve performance on the

Fact 1	I love cats and have two cats
Fact 2	I've a hat collection of over 1000 hats.
Blender	My cats names are all the hats i have
+ Per. Attr.	My cats are called kitties
Fact 1	I am a doctor.
Fact 2	My daughter is a child prodigy.
Blender	My daughter is prodigy so she gets a lot of accidents.
+ Per. Attr.	I've seen a lot of accidents.

Table 2: Examples of incorrect utterances generated by Blender by mixing up two facts.

PersonaChat task. A qualitative analysis of the generated utterances suggests that including personal attributes into Blender can more effectively inform the model which persona statement to focus on during generation. This can prevent Blender from including information from irrelevant persona statements (*e.g.* by mixing up facts from two unrelated persona statements), as shown in Table 2.

4 Personal Attribute Tasks

Having motivated the importance of personal attribute KG triples, we propose the task of obtaining them from natural language sentences. We first explain how we formulate two complementary tasks from DialogNLI data and then formally define our tasks. Finally, we analyze the task datasets to gather insights into the linguistic phenomena that our tasks involve.

4.1 Extraction and Inference Tasks

We formulate two tasks by partitioning the DialogNLI dataset into two non-overlapping subsets, after removing invalid samples with triples containing *None* or <*blank*>. Here, each **sample** refers to a sentence paired with an annotated triple. Train/dev./test splits follow DialogNLI, with descriptive statistics shown in Table 3. The dataset for the EXTRACTION task contains samples in which both the head and tail entities are spans inside the paired sentence. An example is (I, [has_profession], receptionist) from the sentence "I work as a receptionist at a lawyers office". We formulate the EXTRACTION task in a similar way to existing Relation Extraction tasks such as ACE05 (Wadden et al., 2019) and NYT24 (Nayak and Ng, 2020). This allows us to apply modeling lessons learned from an established task formulation.[1]

[1]We considered framing our task as joint intent classification and slot filling. However, the DialogNLI dataset is more suited for Relation Extraction-style formulation because there are always two domain-slots for each relation and domain-slots for each relation are not always unambiguous.

The complementary set is the dataset for the INFERENCE task, for which head entity and/or the tail entity cannot be found as spans within the paired sentence. This is because the head and tail entities are paraphrased from the original sentence or derived based on commonsense reasoning used by annotators. An example of a paraphrased triple is (I, [physical_attribute], tall) from the sentence "I am in the 99th height percentile", while one based on commonsense reasoning is (I, [want_job], commentator) from the sentence "my ultimate goal would be calling a ball game".

	EXTRACTION	INFERENCE
Samples		
train	26759	32031
dev.	2982	3281
test	2881	3249
Unique elements		
head entities	118	143
relations	60	60
tail entities	3558	4487
Avg. words		
head entities	1.02	1.08
relations	1.00	1.00
tail entities	1.26	1.68
sentences	12.9	12.1

Table 3: Statistics of the dataset for the two tasks.

The INFERENCE task is posed as a challenging version of the EXTRACTION task that tests models' ability to identify pertinent information in sentences and then make commonsense inferences/paraphrases based on such information. An existing task has sought to predict personal attributes that are not always explicitly found within sentences (Wu et al., 2019). However, it did not distinguish between personal attributes that can be explicitly found within sentences (*i.e.* EXTRACTION) from those that cannot (*i.e.* INFERENCE) . We believe that, given that the inherent difficulty of identifying the two types of personal attributes are greatly different, it is helpful to pose them as two separate tasks. In this way, the research community can first aim for an adequate performance on the simpler task before applying lessons to make progress at the more challenging task. This is also the first time that personal attributes that are not explicitly contained in sentences are shown to be derivable from words in the sentence using commonsense/lexical inferences.

4.2 Formal Task Definition

Given a sentence S, we want to obtain a personal-attribute triple in the form of (**head entity, relation, tail entity**). The relation must belong to a set of 60 predefined relations. In the EXTRACTION subset, the head entity and tail entity are spans within S. Conversely, in the INFERENCE subset, the head entity and/or the tail entity cannot be found as spans within S.

4.3 Dataset Analysis

We analyze the datasets to obtain insights into how the tasks can be approached. Because the majority of head entities (93.3%) are simply the word "I", our analysis will focus on tail entities.

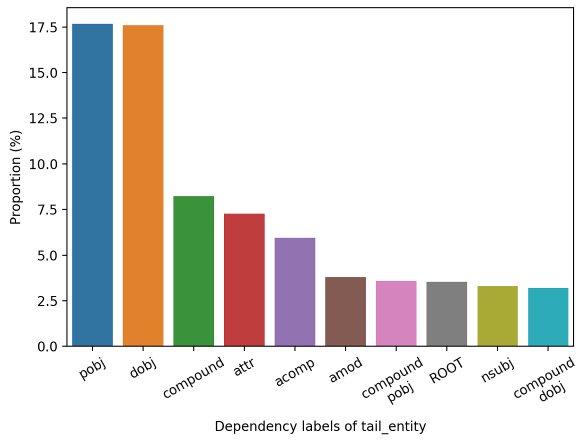

Figure 2: Bar plot for 10 most common dependency labels of tail entities

Dataset for the EXTRACTION task We use dependency parses of sentences to understand the relationship between words within tail entities and the sentence ROOT. Dependency parsing was chosen because it is a well-studied syntactic task (Nivre et al., 2016) and has been shown to contribute to the relation extraction task (Zhang et al., 2017). Dependency parses and labels associated with each dependent word were identified using a pre-trained transformer model from spaCy.[2] The parser was trained on data annotated with the ClearNLP dependency schema that is similar to Universal Dependencies (Nivre et al., 2016)[3]

As shown in Figure 2, objects of prepositions (pobj) and direct objects (dobj) each comprises of 17.5% of tail entities, followed by compound words (compound), attributes (attr) and adjectival complements (acomp), plus 138 other long-tail labels. The range of grammatical roles as well as the fact that one third of tail entities do not involve nouns (see Figure in Section A.2) also suggest that the tail entities in our dataset goes beyond proper nouns, which are what many Relation Extraction datasets (*e.g.*, ACE05 and NYT24) are mainly concerned with. Such diversity in grammatical roles played by tail entities means that approaches based on parsing/named entity recognition alone are unlikely to be successful in the EXTRACTION task.

Dataset for the INFERENCE task A qualitative inspection of the dataset showed that inferences can be made on the basis of paraphrasing/semantically-related words and commonsense inferences, as shown in examples discussed in Section 4.1. To better understand how tail entities can be inferred from the sentence in the INFERENCE subset, we analyze the relationship between words in the tail entity and words in the sentence. 77.7% of tail entities cannot be directly identified in the sentence. We performed a few transformation to identify potential links between the tail entity and the sentence. *ConceptNet_connect* refers to words with highest-weighted edges on ConceptNet to sentence words while *ConceptNet_related* refers to words that have closest embedding distances to sentence words. Details of their preparation are in Appendix A.3. As in Table 4, our analysis shows that a model that can perform well on the INFERENCE task requiring both WordNet semantic knowledge (Fellbaum, 1998) as well as ConceptNet commonsense knowledge (Speer et al., 2017).

Transformation	**Example** (sentence→tail entity)	**%**
ConceptNet_related	mother →female	**65.7**
ConceptNet_connect	wife →married	56.1
WordNet_synonym	outside →outdoors	41.8
WordNet_hypernym	drum →instrument	10.4
WordNet_hyponym	felines →cats	9.92
Same_stem	swimming →swim	35.1

Table 4: Proportion of tail entities that can be related to sentence words after applying each transformation.

[2]https://spacy.io/
[3]https://github.com/clir/clearnlp-guidelines/blob/master/md/specifications/dependency_labels.md

5 GenRe

This section proposes GenRe, a model that uses a unified architecture for both the EXTRACTION and the INFERENCE tasks. We use a generator-reranker framework to address the needs of the two tasks. On one hand, a generative model is necessary because head and/or tail entities cannot be directly extracted from the sentence for INFERENCE dataset. On the other hand, given preliminary experiments on data from the EXTRACTION dataset, we observed that a large proportion of correct triple is among the top-k - but not top-1 - outputs (see Table 8 for details). A reranker can then be used to select the most likely triple among the top-k triples.

5.1 Generator

We use an autoregressive language model (GPT-2 small) as our generator because its extensive pretraining is useful in generating syntactically and semantically coherent entities. The small model was chosen to keep model size similar to baselines. We finetune this model to predict a personal attribute triple occurring in a given input sentence. Specifically, we treat the flattened triples as targets to be predicted using the original sentence as context. The triple is formatted with control tokens to distinguish the head entity, relation, and tail entity as follows:

$\mathbf{y}$ = [HEAD], $t^{head}_{1:m}$, [RELN], t^{reln}, [TAIL], $t^{tail}_{1:k}$ where {[HEAD],[RELN], [TAIL]} are control tokens, $t^{head}_{1:m}$ is the head entity (a sequence of length m), t^{reln} is a relation, and $t^{tail}_{1:k}$ is the tail entity.

During evaluation, we are given a sentence as context and seek to generate a personal attribute triple in the flattened format as above. To reduce the search space, we adopt a constrained generation approach. Specifically, after the [RELN] token, only one of 60 predefined relations can be generated, and so the output probability of all other tokens is set to 0. After the [TAIL] token, all output tokens not appearing in the input sentence will have zeroed probabilities in the EXTRACTION task. Conversely for the INFERENCE task, the only allowed output tokens after the [TAIL] token are those which have appeared following the predicted relation in the training data. For example, tail entities that can be generated with a [physical_attributes] relation include "short", "skinny" or "wears glasses", as these examples occur in the training data. Implementation details are in Appendix A.4.

5.2 Reranker

We use BERT-base as the Reranker because its bidirectionality allows tail tokens to influence the choice of relation tokens and contributes to better commonsense understanding (Zhou et al., 2020). These benefits have led to many relation extraction models using BERT as part of the pipeline (Wadden et al., 2019; Yu et al., 2020; Ye et al., 2021).

For each S, we obtain the L most likely sequences using the Generator, including the context sentence. Each sequence is labelled as correct or incorrect based on whether the predicted triple (head entity, relation, tail entity) matches exactly the ground-truth triple. We fine-tune a BERT model with a binary cross-entropy loss function to classify whether sequences are correct. During inference, we select the sequence with the highest likelihood of being correct as our predicted sequence. We set L to 10 in all experiments. Implementation details are in Appendix A.5.

6 Experiments

We first explain the metrics used in the experiments. Next, we introduce the baseline models. Finally, we examine how GenRe compares to baseline models to understand its advantages and limitations.

6.1 Metrics

Micro-averaged Precision/Recall/F1 were calculated following Nayak and Ng (2020), in which a sample is considered correct only when all three elements (head_entity, relation and tail entity) are resolved correctly. In addition, metrics were calculated for each element of the triple to provide insights into analysis.

6.2 Baseline Models

Generative models can be used for both the EXTRACTION and the INFERENCE tasks.

WDec is an encoder-decoder model that achieved state-of-the-art performance in the NYT24 and NYT29 tasks (Nayak and Ng, 2020). The encoder is a Bi-LSTM, while the decoder is an LSTM with attention over encoder states. An optional copy mechanism can be used: when used, the decoder will only generate tokens found in the original sentence. The copy mechanism was used on the EXTRACTION dataset but not on the INFERENCE dataset (given their better empirical performance).

GPT2 is an autoregressive language model that we build GenRe on. We used the same model configuration as in GenRe.

Extractive models can be used only for the EXTRACTION task, because they select for head and tail entities from the original sentence.

DyGIE++ is a RoBERTa-based model that achieved state-of-the-art performance in multiple relation extraction tasks including ACE05 (Wadden et al., 2019). It first extracts spans within the original sentence as head and tail entities. Then, it pairs up these entities with a relation and passes them through a graph neural network, with the head and tail entities as the nodes, and relations as the edges. This allows information flow between related entities before passing the triple through a classifier.

PNDec is an Encoder-Decoder model that achieved close to SOTA performance in NYT24 and NYT29 (Nayak and Ng, 2020). It uses the same encoder as WDec but uses a pointer network to identify head and tail entities from the original sentence, which it pairs with possible relation tokens to form a triple that is subsequently classified.

All baseline models were trained on our datasets using their suggested hyper-parameters.

6.3 Model Results

	EXTRACTION			INFERENCE		
	P	R	F1	P	R	F1
GenRe	49.4	38.1	43.0	**29.5**	**25.9**	**27.5**
Generative						
WDec	46.4	39.9	42.9	20.8*	21.8*	21.3*
GPT2	46.5*	26.7*	33.9*	19.3*	13.1*	15.6*
Extractive						
DyGIE++	48.6	**40.3**	44.1			
PNDec	**50.4**	39.8	**44.5**			

Table 5: Performance on the test set. *Significantly different from GenRe with 5 runs based on a two-tailed t-test (p<0.01). All other differences are not significant.

The top-performing baseline models on the EXTRACTION dataset are the extractive models, which select spans within the sentence and classify whether an entire triple is likely to be correct. Generative models perform more poorly; generation happens in a left-to-right manner meaning that some elements of the triple have to be generated without knowing what the other elements are. Our Reranker alleviates some of this drawback but does not overcome this problem entirely; our performance yields better precision than the other generative models, and comes within 1.5 F1 score points of the top performing extractive model. On the other hand, extractive models cannot solve the INFERENCE task, because the underlying assumption that head and tail entities must be found within the sentence does not hold. When compared to baseline generative models that can be applied to the INFERENCE task, our model performs significantly better (by ≥ 6.2 F1 points).

7 Analysis

We first conduct ablation studies to better understand the contribution of constrained generation and the Reranker, by measuring the performance of our model when each component is removed. Then, we seek to understand individual elements of predicted personal attribute triple to identify future areas of improvement. We focus on the relation field (F1 = 42.2 to 47.8%) as well as the tail entity (F1 = 38.1 to 61.4%) because they contain more errors than the head entity (F1 = 86.3 to 97.2%). Specifically, we analyze the misclassification of relations in the EXTRACTION task as well as the extent to which the predicted tail entities in the INFERENCE task demonstrate commonsense and semantic understanding.

7.1 Ablation Study

Table 6 shows that both the Reranker and constrained Generation contribute to the performance of GenRe. In particular, the Reranker plays a more significant role on the EXTRACTION dataset while the constrained Generation plays a greater role on the INFERENCE dataset.

	EXTRACTION			INFERENCE		
	P	R	F1	P	R	F1
GenRe	49.4	38.1	43.0	29.5	25.9	27.5
- Constr. Gen	47.8	33.7	39.5	23.2	18.0	20.3
- Reranker	45.1	29.3	35.5	28.0	23.6	25.6

Table 6: Ablation study for Reranker and constrained generation.

Constrained generation has modest impact on the EXTRACTION dataset, likely because the generator can learn to limit its generation to spans from the context sentence. However, on the INFERENCE dataset, the original search space cannot be effectively limited to tokens in the context sentence. Therefore, applying the heuristic that only

					Top 3 Most Frequent (n)		
Dataset	True Relation (n)	P	R	F1	Predicted Relations	True Tail Entities	Predicted Tail Entities
EXTRACTION	[like_activity] (191)	34.3	20.5	25.7	[like_activity] (89) [has_hobby] (34) [like_general] (20)	singing (24) mechanics (13) video games (11)	singing (11) fixing (9) traveling (9)
	[have_children] (162)	65.2	48.3	55.5	[have_chidren] (122) [marital_status] (11) [other] (7)	children (53) son (26) 2 children (7)	2 children (13) children (9) 5 son (9)
INFERENCE	[has_hobby] (254)	38.4	28.6	32.8	[like_activity] (85) [has_hobby] (73) [has_profession] (13)	volunteer (36) ping pong (14) skateboarding (13)	volunteer (36) rock climbing (16) skateboarding (13)
	[has_profession] (237)	72.5	59.5	65.4	[has_profession] (158) [employed_by_general] (11) [has_ability] (8)	musician (19) chef (13) electrician (13)	musician (16) chef (13) electrician (12)

Table 7: Some common relations in EXTRACTION and INFERENCE datasets

tail entities associated with a particular relation (in the training set) can be decoded is useful. This is particularly helpful for relations appearing with few possible tail entities such as [have_children], shown in Table 7.

The Reranker is needed because, many times, the correct triple for samples in the EXTRACTION task can be generated by the Generator but might not be the triple that is predicted to have the highest likelihood, as shown in Table 8. In particular, the predicted most likely relation and the tail entity can be partly correct ("old"/"retiring" instead of "60"). This suggests the need to refine the generated outputs of the Generator using a Reranker. The maximum possible recall on the EXTRACTION TASK increases from 29.5% to 58.1% when considering 10 instead of only 1 generated candidate. While the achieved recall (38.1%) is far from the maximum possible recall, the achieved recall is still significantly higher than using the Generator alone. On the other hand, the Reranker plays much lesser of a role on the INFERENCE task because generating the correct triple within the top 10 candidates is more of a bottleneck in this task.

Utterance	How old are you? I am 60, retiring in a few years.		
Elements	head entity	relation	tail entity
Ground truth	I	[has_age]	60
Predicted	I	[has_age]	old
Most likely	I	[has_age]	60
↓	I	[has_age]	retiring
	I	[other]	old
Less likely	I	[want_do]	retiring

Table 8: Example of Generator output (Extraction dataset) to demonstrate role of the Reranker.

7.2 Misclassification of Relations

Major sources of error on the EXTRACTION dataset came from relation tokens that have close semantic meanings. They were either synonyms of one another (*e.g.*, [has_hobby] vs [like_activity]) or hyponyms/hypernyms of one another (*e.g.*, [like_general] vs [like_activity]), as illustrated in Table 7. Such errors likely arose due to the way that the DialogNLI dataset (Welleck et al., 2019) was annotated. Specifically, annotators were asked to label a single possible triple given a sentence instead of all possible triples. Because of this, our evaluation metrics are likely to over-penalize models when they generated reasonable triples that did not match the ground truth. Future work can avoid this problem by labelling all possible triples and framing the task as multilabel learning.

7.3 Commonsense and Semantic Understanding in Tail Entities

We analyzed tail entities that were predicted by each models in the INFERENCE dataset, as a proportion of ground-truth transformations in Table 4. As shown in Table 9, GenRe performed better on transformations involving lexical semantics (particularly WordNet hypernyms and hyponyms) while WDec performed better on ConceptNet connected and related words, demonstrating better commonsense inference ability. This is likely because GenRe generation was constrained to tokens from tail entities that were associated with relations in the training set. This increased the prediction of words from similar semantic categories (*e.g.* jobs or hobbies) but reduced the prediction of words that are not of the same semantic category but were linked via commonsense reasoning. Nonetheless, only a maximum of 57.2% of transformations were captured, suggesting that further modeling improvements are necessary to capture lexical semantics and commonsense understanding as required in this task.

Transformation	GenRe	WDec
ConceptNet_related	50.6	57.2
ConceptNet_connect	47.0	56.4
WordNet_synonym	35.9	48.5
WordNet_hypernym	32.6	27.2
WordNet_hyponym	33.5	28.5
Same_stem	36.2	40.4

Table 9: Proportion of transformations captured by predicted tail entities of models on the INFERENCE task

8 Related Work

Applications of Personal Attributes: Personal attributes can be used to ground open-domain chit-chat dialogue agents to minimize inconsistencies in their language use (*e.g.*, **I like cabbage** →(next turn) →**Cabbage is disgusting**) (Majumder et al., 2020; Li et al., 2016; Mazaré et al., 2018; Qian et al., 2018; Zheng et al., 2020a,b). Such grounding can also give dialogue agents more captivating personalities that make them engaging to talk to (Zhang et al., 2018; Li et al., 2020). Thus far, personalization in chit-chat has made use of dense embeddings and natural language sentences. While KG triples have been shown to be capable of grounding Natural Language Generation (Moon et al., 2019; Koncel-Kedziorski et al., 2019), they have yet to be used in the context of dialogue personalization.

Personal attributes can also ground task-oriented dialogue agents to provide more personalized recommendations (Joshi et al., 2017; Luo et al., 2019; Pei et al., 2021; Lu et al., 2019; Mo et al., 2017). However, such personalized recommendations have thus far been made only with a small set of one-hot features (< 50) for single-domain tasks. While personalization in open-domain task-oriented settings remains unexplored, it is likely to involve orders of magnitude more features, thus making KG triples more suitable to select and utilise pertinent features.

Personal Attribute Extraction: Most work on extracting personal attributes from natural language (Tigunova et al., 2019, 2020; Pappu and Rudnicky, 2014; Mazaré et al., 2018; Wu et al., 2019) employed distant supervision approaches using heuristics and hand-crafted templates, which are known to have poor recall. In contrast, we use a strong supervision approach in which triples were manually annotated. Yu et al. (2020); Li et al. (2014) also attempted to extract personal information from dialogue using a strongly supervised paradigm. However, they focused on demographic attributes as well as interpersonal relationships, which contrast with our focus on what people own and like. Yu et al. (2020) used BERT to identify relations between given entities while Li et al. (2014) used SVMs to classify relations and CRFs to perform slot filling of entities.

Generating KG triple-like Data using Language Models: Autoregressive language models have been applied to a wide range of tasks involving the generation of data with similar structures as personal attribute KG triples, including dialogue state tracking (Hosseini-Asl et al., 2020) and commonsense KG completion (Bosselut et al., 2019). The most similar application is Alt et al. (2019), which used the original GPT model (Radford and Narasimhan, 2018) for relation classification. Their task formulation involves identifying a specific relation (out of around 30 possible options) for two given entities. On the other hand, our tasks seek to identify not only the relation, but also the head and tail entities, which have potentially open vocabulary requirements that significantly increases the search space.

9 Conclusion

In conclusion, we demonstrate that personal attributes can support the personalization of open-domain social chit-chat agents. Thus motivated, we propose novel tasks of extracting and inferring personal attributes from dialogue and carefully analyze the linguistic demands of these tasks. To meet the challenges of our tasks, we present GenRe, a model which combines constrained attribute generation and re-ranking on top of pre-trained language models. GenRe achieves competitive performance vs. established Relation Extraction baselines on the EXTRACTION task (within 1.5 F1 points). On the more challenging INFERENCE task that involve lexical and commonsense inferences, our model outperforms Relation Extraction baselines by 6.2 F1 points. Together, our work contributes an important step towards realizing the potential of personal attributes in personalization of dialogue agents.

Ethics and Broader Impact

Privacy in real world applications Because our task involves extracting and inferring personal attributes, real-world users should be given the option to disallow particular types of relations from being collected and/or used for downstream applications. Users should also be given the freedom to delete their collected personal attributes. A further step might be to restrict the extraction and storage of personal attributes to only local devices using differential privacy and federated learning techniques.

References

Christoph Alt, Marc Hübner, and Leonhard Hennig. 2019. Improving relation extraction by pre-trained language representations. In *Automated Knowledge Base Construction (AKBC)*.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. 2019. The second conversational intelligence challenge (convai2).

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. 2021. Knowledge graphs.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue.

Chaitanya K. Joshi, Fei Mi, and Boi Faltings. 2017. Personalization in goal-oriented dialog.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text Generation from Knowledge Graphs with Graph Transformers. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2284–2293, Minneapolis, Minnesota. Association for Computational Linguistics.

Aaron W. Li, Veronica Jiang, Steven Y. Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020. Aloha: Artificial learning of human attributes for dialogue agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8155–8163.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. 2016. A persona-based neural conversation model. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 994–1003, Berlin, Germany. Association for Computational Linguistics.

X. Li, G. Tur, D. Hakkani-Tür, and Q. Li. 2014. Personal knowledge graph population from user utterances in conversational understanding. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 224–229.

Yichao Lu, Manisha Srivastava, Jared Kramer, Heba Elfardy, Andrea Kahn, Song Wang, and Vikas Bhardwaj. 2019. Goal-oriented end-to-end conversational models with profile features in a real-world setting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 48–55, Minneapolis, Minnesota. Association for Computational Linguistics.

Liangchen Luo, Wenhao Huang, Qi Zeng, Zaiqing Nie, and Xu Sun. 2019. Learning personalized end-to-end goal-oriented dialog. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6794–6801.

Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. 2020. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9194–9206, Online. Association for Computational Linguistics.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

A. H. Miller, W. Feng, A. Fisch, J. Lu, D. Batra, A. Bordes, D. Parikh, and J. Weston. 2017. Parlai: A dialog research software platform. *arXiv preprint arXiv:1705.06476*.

Kaixiang Mo, Shuangyin Li, Yu Zhang, Jiajun Li, and Qiang Yang. 2017. Personalizing a dialogue system with transfer reinforcement learning.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. 2019. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy. Association for Computational Linguistics.

Tapas Nayak and Hwee Tou Ng. 2020. Effective modeling of encoder-decoder architecture for joint entity and relation extraction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8528–8535.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal Dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

Aasish Pappu and Alexander Rudnicky. 2014. Knowledge acquisition strategies for goal-oriented dialog systems. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 194–198, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Jiahuan Pei, Pengjie Ren, and Maarten de Rijke. 2021. A cooperative memory network for personalized task-oriented dialogue systems with incomplete user profiles. *arXiv preprint arXiv:2102.08322*.

Martin F Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Qiao Qian, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Assigning personality/profile to a chatting machine for coherent conversation generation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4279–4285. International Joint Conferences on Artificial Intelligence Organization.

A. Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M. Smith, Y-Lan Boureau, and Jason Weston. 2020. Recipes for building an open-domain chatbot.

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*, AAAI'17, page 4444–4451. AAAI Press.

Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. 2019. Listening between the lines: Learning personal attributes from conversations.

Anna Tigunova, Andrew Yates, Paramita Mirza, and Gerhard Weikum. 2020. CHARM: Inferring personal attributes from conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5391–5404, Online. Association for Computational Linguistics.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span 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)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Sean Welleck, Jason Weston, Arthur Szlam, and Kyunghyun Cho. 2019. Dialogue natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3731–3741, Florence, Italy. Association for Computational Linguistics.

Chien-Sheng Wu, Andrea Madotto, Zhaojiang Lin, Peng Xu, and Pascale Fung. 2019. Getting to know you: User attribute extraction from dialogues. *arXiv preprint arXiv:1908.04621*.

Hongbin Ye, Ningyu Zhang, Shumin Deng, Mosha Chen, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Contrastive triple extraction with generative transformer.

Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940, Online. Association for Computational Linguistics.

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 (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia. Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.

Yinhe Zheng, Guanyi Chen, Minlie Huang, Song Liu, and Xuan Zhu. 2020a. Personalized dialogue generation with diversified traits.

Yinhe Zheng, Rongsheng Zhang, Minlie Huang, and Xiaoxi Mao. 2020b. A pre-training based personalized dialogue generation model with persona-sparse data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9693–9700.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9733–9740.

A Appendix

A.1 Blender Fine-tuning Details

Finetuning hyperparameters are taken from https://parl.ai/projects/recipes/, with the exception of validation metric changed to Hits@1. Each fine-tuning epoch takes 1.5 hours on a Nvidia V100 GPU. We only prepend personal attributes before system utterances but not user utterances. Metrics are for the validation set because test set was not available. All experiments were conducted using ParlAI (Miller et al., 2017).

A.2 Task Analysis Details

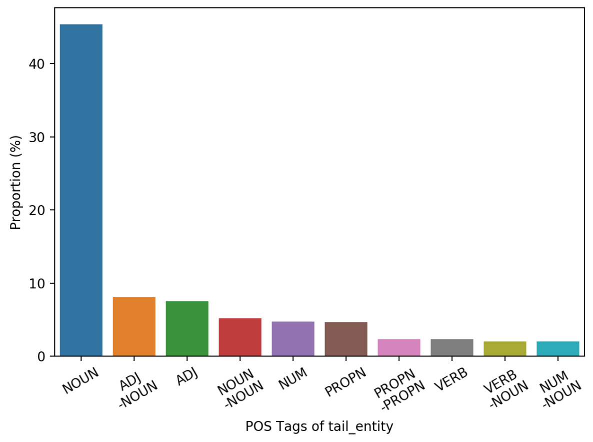

Figure 3: Bar plot for 10 most common POS tags of tail entities.

A.3 Details of Transformations to Link Tail Entity to Sentence

ConceptNet_related: All words in the tail entity can be found in the 100 most related words to each sentence word based on embedding distance on ConceptNet

ConceptNet_connect: All words in the tail entity can be found in the 100 words that have the highest-weighted edge with each sentence word on ConceptNet.

WordNet_synonym: All words in the tail entity can be found in the synonyms of every synset of each sentence word on WordNet.

WordNet_hypernym: All words in the tail entity can be found in the hypernyms of every synset of each sentence word on WordNet

WordNet_hyponym: All words in the tail entity can be found in the hyponyms of every synset of each sentence word on WordNet

Same_stem: All words in the sentence and tail entity are stemmed using a Porter Stemmer (Porter, 1980) before searching for the tail entity in the sentence

A.4 Generator Details

GPT-2-small was used. Additional special tokens including the control tokens ([HEAD], [RELN], [TAIL]) as well as relation tokens were added into the tokenizer. Beam search decoding (beam size = 10) was used at inference time. GPT2-small was accessed from HuggingFace Transformers library with 125M parameters, context window 1024, 768-hidden, 768-hidden, 12-heads, dropout = 0.1. AdamW optimizer was used with $\alpha = 7.5 * 10^{-4}$ for the EXTRACTION dataset and $\alpha = 2.5 * 10^{-3}$ for the INFERENCE dataset, following a uniform search using F1 as the criterion at intervals of $\{2.5, 5, 7.5, 10\} * 10^{n}; -5 \leq n \leq -3$. Learning rate was linearly decayed (over a max epoch of 8) with 100 warm-up steps. Each training epoch took around 0.5 hour on an Nvidia V100 GPU with a batch size of 16. Validation was done every 0.25 epochs during training.

A.5 Reranker Details

BERT-base-uncased was used. Additional special tokens including the control tokens ([HEAD], [RELN], [TAIL]) as well as relation tokens were added into the tokenizer. BERT-base-uncased was accessed from HuggingFace Transformers library (with 12-layer, 768-hidden, 12-heads, 110M parameters, dropout = 0.1). The choice of the base model was made to have fairness of comparison with baseline models in terms of model size. AdamW optimizer was used with $\alpha = 5 * 10^{-6}$, following a uniform search using F1 as the criterion at intervals of $\{2.5, 5, 7.5, 10\} * 10^{n}; -6 \leq n \leq -3$. Learning rate was linearly decayed (over a max epoch of 8) with 100 warm-up steps. Each training epoch took around 1 hour on an Nvidia V100 GPU with a batch size of 10.Validation was done every 0.25 epochs during training.

A.6 Model Validation Performance

	Extraction			Inference		
	P	R	F1	P	R	F1
GenRe	49.1	**38.8**	43.3	**27.4**	**22.7**	**24.8**
Generative						
WDec	46.1	37.9	41.6	16.2	18.6	17.3
GPT2	46.7	27.8	34.8	21.0	15.7	18.0
Extractive						
DyGIE++	49.4	34.9	40.9			
PNDec	**51.3**	38.1	**43.8**			

Table 10: Performance on the validation set