On evaluating dialog models and choosing negative samples

Théo Matussière, École polytechnique
theo.matussiere@polytechnique.edu

September 18, 2018

Abstract

Dialog is arguably the most human trait of intelligence we can find. The vision problems like object recognition are related to intelligence, but we seem to share these skills with most living animals. In that perspective the Artificial Intelligence community strives to model and understand dialog. These dialog models, colloquially known as chatbots, have enjoyed a fast pace of development in the recent years with the parallel development of reliable sentence encoders.

Dialog models exist in two competing ways; generative methods and ranking algorithms. In this work we will explore the current state of the art in ranking algorithms and extend some works in their conditioning and the way we evaluate them. We propose new metrics for the dialog task and show some empirical results on choices made during training on the composition of batches.

The reader up to date with the literature in dense representations and the dialog task can skip the first section, although subsection 1.3 contains original work and will be referred to in later sections.

Contents

1 Problem description & current solutions **3**
1.1 Dense representations of words and sentences . . . 3
1.2 The Transformer and learning to converse by ranking sentences . . . 5
1.3 Formalizing dialog . . . 7

2 Incremental works **10**
2.1 The conditioned Transformer . . . 10
2.2 Exploring ranking methods . . . 11
2.3 Different mixing strategies . . . 12
2.4 Limitations of ranking based agents & conversation history . . . 13

3 Original works: batching **15**
3.1 Noise contrastive estimation & negative sampling . . . 15
3.2 The Reddit dataset . . . 15
3.3 Shuffling or not, introspecting distribution on sentences and batch composition 16

4 Original works: Metrics, DialogEval & new datasets **19**
4.1 Gathering human evaluations . . . 19
4.2 Extending ranking tasks . . . 20
4.3 The DialogEval toolkit . . . 22
4.4 Natural Language Inference . . . 24

Thanks

This internship was made possible by Pierre-Emmanuel Mazaré, who has been a dedicated and inspiring mentor throughout these four months. I would like to thank him warmly, as well as Samuel Humeau, who also works as an engineer in the team, and with whom I've had the pleasure of working as he was available and helpful. Their feedback and our discussions shaped this work.

On the other side of the Atlantic in the New York City offices were Jason Weston, Emily Dinan and Jack Urbaneck, who were of great help and guidance. I've also the chance to meet Cathal O'Madagain, thanks to Marco Baroni, who has been of great help in formalizing and conceptualizing some aspects of this work.

This experience would not have been the same without my fellow interns Amaury Sabran, Jean Tarbouriech, Jean Alaux–Lorrain and Jules Pondard. Our daily trainee catchup meeting are fond memories.

1 Problem description & current solutions

Dialog models have existed in some form or another since the sixties, with notable examples being ELIZA (Weizenbaum 1966) and SHRDLU (Winograd 1971). Both deserve some attention so that we can properly define our intent with this work. ELIZA was an algorithm that mimicked a therapist, defaulting to questions relating to the emotions of its dialog partner. SHRDLU is not exactly a dialog model as much as it is a language interface for a very simple game in which you describe actions in natural language to the algorithm.

In our current denominations we would dub them "goal oriented" dialog models. Goal oriented dialog models imply a limit to the space of possible answers, i.e. booking an appointment with a doctor. Limiting this space of possibles made it possible to deal with hard coded rules and simple syntax forms. Gradually we can extend these goals to more complex environments, as demonstrated by Google with their Duplex system[1].

However we are here concerned with a more general setting for conversations. This setting is the chit-chat type of conversation, i.e. a discussion where:

1. the conversation is short, less than 10 rounds of dialog,
2. the topics involved are *light* i.e. not emotional or too involving for the dialog partners, unimportant,
3. the conversation is *shallow*, meaning that no topic is talked about for too long or into its depths.

As for all of the Natural Language Processing (NLP) field, the 2013 year marked a big leap, with chatbots having made little progress in the decades before that. Before that methods drew on insights from information retrieval systems, with the ubiquitous use of the Term Frequency-inverse Decrease Frequency (TFiDF, Salton and Buckley 1988) method, and other methods from Markov fields and latent spaces modelization.

A big leap was made in 2013 with the advent of dense representations of words that enable us to encode meaning into real-valued vectors, i.e. in metric spaces. This started with meaning at the scale of the word and gradually shifted to sentences, though the challenge remains open.

We'll explore these representations in the following sections as they are essential to the nature of the algorithm we develop.

1.1 Dense representations of words and sentences

Word vectors

Here we will briefly present the key findings of the Word2Vec algorithm (Mikolov, Sutskever, et al. 2013), and we will see the first instance of a training scheme it popularized and that we heavily use in our dialog model; negative sampling.

Word2Vec tries to learn a continuous mapping between semantic similarity and a vector space; it is a function $\phi : D \to \mathbb{R}^d$ with D a dictionary of words. Our natural definition of continuity demands a distance on D, which is taken to be the natural semantic similarity between two words[2]: CAT is close to KITTEN, which is far from WAR. The idea behind Word2Vec is to bypass this intangible metric and to learn this mapping with the distributional hypothesis' heuristic, as stated in Firth 1957: "You shall know a word by the company it keeps.".

[1] See https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html

[2] Although it is ill-defined as a mathematical distance since polysemic words imply violating the triangular inequality.

This is embodied by the objective function of the training algorithm exposed in equation 1: at each time step we move from word to word in a corpus (wikipedia or web crawls), so that at time t a word w is considered.

$$\sum_{c \in N_k(w)} \log \sigma\big(\phi\,(c) \cdot \phi\,(w)\,\big) + \sum_{c' \in F_{k'}(w)} \log \sigma\big(-\phi\,(c') \cdot \phi\,(w)\,\big) \tag{1}$$

This word occurs in a sentence with some preceding and some succeeding words, forming a neighborhood $N_k(w)$ where k is a parameter ruling the width of the neighborhood. We then sample n other words randomly, according to some distribution — taken to be the unigram distribution raised to the power 3/4 in Mikolov, Chen, et al. 2013, forming a set of negative samples $F_{k'}(w)$. In equation 1 we noted σ as the sigmoid function: $x \mapsto (1 + e^{-x})^{-1}$.

Improvements have been made to the Word2Vec algorithms, chiefly by the GloVe embeddings (Pennington, Socher, and Manning 2014) and the fastText ones (Bojanowski et al. 2016) which we use in our encoders.

Sentence vectors

Now equipped with word vectors that are working well and exhibiting nice properties such as the much publicized forms of semantic algebra (king - man + woman $\sim$ queen), we can begin to contemplate representing whole sentences as vectors too. However the semantic space for sentences is richer and more complex than for the words. A simple thought experiment shows this simply; try picturing the size of the set of all acceptable sentences of only three words. This space will include 'How are you?', 'I like bees.', 'Away with you!', etc.... It is huge. And this is only for fixed-size sentences, so that the actual space of all possible sentences in any language is unfathomable when we take into account the recursive nature of language.

So how do we do this? There has been a lot of different ways to do this. The naive way is of course to take the mean of the word vectors in the sentence, or a *weighted* sum since we know not all words have equal importance. Denoting a sentence s of length n as a sequence of words $s = (w_i)_{0<i\leq n}$, we can introduce the Bag of Words (BOW) encoder:

$$\psi : (w_i)_{0<i\leq n} \longmapsto \frac{1}{n} \sum_i \phi\,(w_i),$$

This was considered as an under-performing baseline for a while and has now been given more attention with the learning of proper weights. As presented in Arora, Liang, and Ma 2017, we can also leverage the TFiDF method to provide with informed weights.

$$\psi\big(\,(w_i)_{0<i\leq n}\,\big) = \frac{1}{n} \sum_i \log \left(\frac{|\text{Corpus}|}{|\{w_i \in \text{Corpus}\}|} \right) \phi\,(w_i).$$

But this loses all the information on the order in which the words appear, so that two sentences with the same words but arguably different meaning will be squashed together.

Sentences being sequential in nature, the most popular architecture has been Recurrent Neural Networks (RNNs), in particular its Long Short Term Memory (LSTM, Hochreiter and Schmidhuber 1997) variant. We won't describe these in too much detail since our work don't make a lot of use of them.

Another performing architecture is the Convolutional Neural Network (CNNs) which was in the past one of the more performing architecture for computer vision. CNNs have the advantage of being highly parallelizable, making them faster to train, and they also are easier to train, the LSTMs requiring a lot of fine-tuning.

1.2 The Transformer and learning to converse by ranking sentences

We've seen in the previous section that reweighting the sum of vectors can have a dramatic impact. The attention mechanism does exactly that and provides a way to learn these weights. It is simple to describe its general form, but in practice the literature is full of tricks and variants for it. We'll offer here the most general form as presented in Vaswani et al. 2017: considering a matrix $M \in \mathbb{R}n, d$ storing n representations of dimension d we will devise a collection of real weights $(\alpha_i)_{0<i\leq n}$ so as to perform $\sum_i \alpha_i M_i$. Attention is computing these α_i as:

$$\alpha = \sigma_{\sqrt{d}}\left(M \cdot K\right),$$

for some $K \in \mathbb{R}^{d,n}$ that can be fixed or learnable via gradient descent, and σ being the softmax function with temperature:

$$\sigma_\tau : x \in \mathbb{R}^d \longmapsto \left[\frac{e^{x_i/\tau}}{\sum_j e^{x_j/\tau}}\right]_{0<i\leq d}. \tag{2}$$

In the future when talking about attention in the context of the Transformer we will omit the temperature for the softmax as it will always be set to $\sqrt{d}$. Hence, writing the weighted sum takes the form of, with V can be taken to be M — which is then a Memory Network, but we will see that later on:

$$\sigma\big(M \cdot K\big) \cdot V$$

This attention mechanism is extremely powerful and has helped boost scores of the LSTM systems in a wide range of tasks including translation and language models.

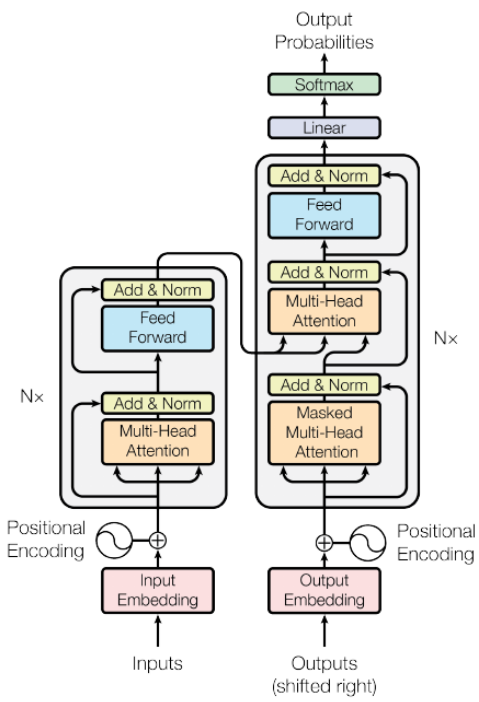

Figure 1: The Transformer architecture as described in Vaswani et al. 2017

The Transformer architecture presented in Vaswani et al. 2017 uses it to its maximum, with the variant of the "self-attention" mechanism, i.e. one where both K and V are M,

up to some transformation with the W_* matrices:

$$\text{Self-Att}\,(M) = \sigma\big(M \cdot W_q \;\cdot\; (M \cdot W_k)^T\big) \;\cdot\; M \cdot W_v$$

This self-attention is done several time in parallel with different W_* matrices, constituting as many "heads" and a module we call Multi-Head Attention. The rest of the network is exposed by figure 1, where we can see the prominence of the attention mechanism. We can also observe the introduction of positional embeddings that addresses the lack of position information during attention.

Our work builds upon this Transformer as it was shown to be state of the art in translation tasks and is fast to compute since it is easily parallelizable.

We're now considering how to effectively build dialog models that *act*; dialog agents. The question becomes how to produce utterances in context with a set of previous ones so that it would make sense in the dialog. This production can be done dynamically, i.e. choosing word after word up until the utterance is deemed complete, or statically by choosing an utterance among a certain number of candidates.

The first possibility corresponds to the range of works using RNNs and the Seq2Seq framework (Sutskever, Vinyals, and Le 2014) like Sordoni et al. 2015 and subsequent articles. These dialog agents perform well under certain regards, but often default to short and non committing answers, making for bland and non engaging conversations. This engagingness problem is an interesting area of research, with efforts like Li, Galley, Brockett, Spithourakis, et al. 2016 which introduces the idea of *persona* which we build upon, Li, Galley, Brockett, Gao, et al. 2015 which proposes modifications to the decoder loss to favor richer answers, or Ghazvininejad et al. 2017 which uses ontologies to inform and ground the answers.

The second somewhat simplistic way of making dialog models is to have a stock of adequate answer candidates and outputting the most appropriate one to the current discussion. This alleviates the vagueness problem encountered by dynamic generation of answers and also settles the grammaticality problem, while of course introducing new kinds of limitations that we will cover in the next subsection. A nice side effect it also has is that provided with candidates in two different languages and a satisfying encoder, the agent can converse in the two languages.

In the most general form, this implies an encoding of candidates, i.e. a mapping $\phi : s \mapsto v \in \mathbb{R}^d$ and an encoder for the context and/or dialog history, and the stimulus (the last utterance), i.e. a mapping $\psi : (s, c) \in \{\text{sentences}\} \times \{\text{contexts}\} \mapsto v' \in \mathbb{R}^d$. Then assuming m candidate replies, encoded by ϕ in a matrix $R \in \mathbb{R}^{m,d}$, for a given context utterance we simply compute:

$$\begin{aligned} \psi\,(s, c) \cdot R &= r \in \mathbb{R}^m \\ i^* &= \underset{0<i\leq m}{\arg\max}\, r_i, \end{aligned}$$

and output the i^*th candidate reply. Note that we take here the inner product as a measure for similarity, but in all generality we ought to describe the ranking process as picking the reply that maximises:

$$\text{sim}\big(\psi\,(s, c)\,, r_i\big),$$

where sim can be the inner product, the cosine similarity, or any positive function $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. Arguably any of the pretrained encoders presented in 1.1 can suffice for this task, though we need to train these representations for the dialog task.

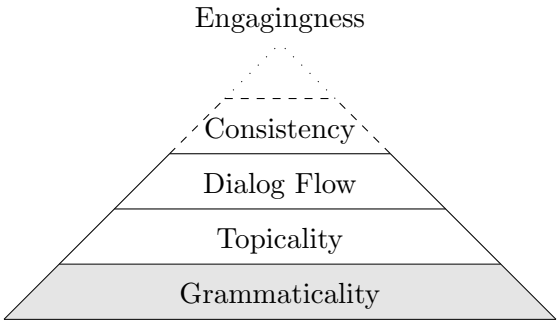

Figure 2: The dialog skills pyramid.

1.3 Formalizing dialog

We present here a minimal conceptual framework for the dialog task. It helps us define how we envision it, which reveals crucial to devise the dialog metrics we present later in section 4.2.

In formalizing dialog, one often finds themselves desperate for tools or landmark works to cling to when searching in the deep learning literature. Our method in formalizing dialog was to inspect modes of failure: what categories can we form of all cases of obvious failure we encounter with our current models? These modes of failure are reminiscent of Grice's maxims (Grice 1975), although the intersection between the ours and theirs is limited since the maxims are hard to automatically assess (and were not meant to be). We recall these maxims here:

1. The maxim of quantity, where one tries to be as informative as one possibly can, and gives as much information as is needed, and no more.

2. The maxim of quality, where one tries to be truthful, and does not give information that is false or that is not supported by evidence.

3. The maxim of [relevance], where one tries to be relevant, and says things that are pertinent to the discussion.

4. The maxim of manner, when one tries to be as clear, as brief, and as orderly as one can in what one says, and where one avoids obscurity and ambiguity[3].

Our conceptualization is as follow with the list of such modes of failure, ordered by difficulty or abstractiveness such as can be seen in figure 2. The concepts here defined are somewhat porous, with examples that could be categorized as one or the other depending on interpretation. We acknowledge these limitations and have worked towards limiting the number of ambiguous cases.

Grammaticality

In the case of this work, which focuses on ranking candidate answers, grammaticality is not an issue since we choose our candidate answers. We did have human evaluators complaining about the level of language in some of our candidates at some point but this was resolved with a stricter filtering of the candidates.

For generative models grammaticality is the first problem to tackle, although it is now a challenge that's well understood thanks to ever improving language models. For these reasons this first obstacle is greyed out in figure 2.

[3]These definitions originated from https://www.sas.upenn.edu/~haroldfs/dravling/grice.html

Topicality

Id	Context	Answer	Topical	Dialog Flow
1	When did your plane land?	It landed at 3pm.	yes	yes
2	When did your plane land?	We were at Roissy Airport.	yes	no
3	When did your plane land?	My cat puked on the couch.	no	no
4	When did your plane land?	The same time as John's	no	yes

Table 1: Examples for the topicality & dialog flow notions.

Topicality is the ability of a dialog partner to answer by respecting the topic of the conversation, which is not necessarily unique. Failing on topicality is commonly designed as producing a *non sequitur*, which is often used for comedic purposes as it suggests a character's aloofness or lack of interest in the ongoing discussion. While this can be considered good human dialog for a number of reasons, we don't expect this kind of behavior from dialog agents, hence remaining a failure case.

Providing a good definition for topicality is not so easy; what to do with example 4? Our solution is to rely on *apparent* topicality which we take to be the presence of words belonging in the same lexicons as the ones present in the previous utterance. In this sense topicality is close to the maxim of relevance we exposed earlier.

However even mastering topicality doesn't make for a good conversation, as we can see by looking at the second example of table 1.

Dialog Flow

We use the term of dialog flow to distinguish with the notion of *dialogical syntax* of Du Bois 2014, which is a whole more thought out than ours, but also more focused on the actual word interplay between contexts and replies in a framework that is closer to syntactical analysis.

There are a few examples of what we intend to designate with dialog flow, that are reminiscent of the philosophical speech-acts (John R Searle and John Rogers Searle 1969) or of the language games of Wittgenstein 1953. The idea is that any context utterance severely constrains the space of possible answers. The basic instance of this is with "yes/no" questions, i.e. questions in the form of MODAL VERB + PRONOUN + ... as in "Did you finish your report yet?". There's also all the types of "wh*" questions i.e. questions starting with what, where, when, who, how, etc...

A limitation to these notions of constraints on the space of possible answers are that in a dialog we can disregard the previous utterance, e.g. with *non sequitur*s to denote disinterest or urgency. This is illustrated with example three of table 1: to some extent this example could be rated as a good dialog interaction, e.g. if we assume the person owns a cat they could make this utterance to signify they didn't board the plane.

This is in fact down to the notion of *shared knowledge*, the idea that between speakers exists a common set of beliefs and knowledge, known and acknowledged by both, that enables shortcuts in the dialog (Davidson 1982) — this discussion was also made in Gundel 1985.

Consistency

Our opinion is that solving consistency is the hardest part of the dialog task; it requires the agent to be *grounded* in the reality of its dialog partner. In Davidson 1982 we find the dialog triangulation concept, tying two dialog partners and their shared world (knowledge); in figure 3 we expose how we see the consistency problem, i.e. as the dotted arrow pointing

from the world to the bot. Since we don't expect our bot to have the ability to modify the world as the human does, the arrow has only one direction.

Consistency is the idea that if the dialog agent states "I like chocolate", it will fail if it later states "I hate chocolate". As for *non sequitur*s in topicality, even though ambivalence is acceptable in humans, it is not a desirable trait for a dialog agent. Whether this existence of fixed relations with an environment constitutes an *identity* is an interesting question, although one we won't develop.

Consistency is also not limited to self-consistency, i.e. if a human tells an agent it is studying at school and further down the bot asks what school the human teaches, we have a consistency problem. In that sense it is also the ability to reason on previous statement and to tie it to external knowledge, which is a task we don't know how to solve yet, hence the dashed lines in figure 2.

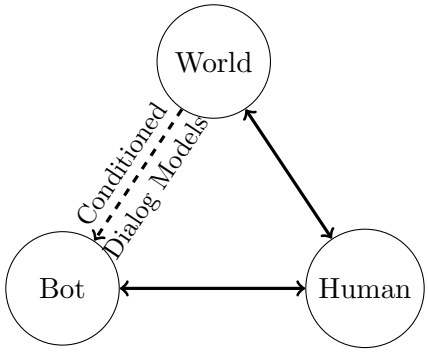

Figure 3: The Davidson Triangulation (Davidson 1982) for Dialog Models

By Conditioned Dialog Models (CDM) we understand all dialog models where the outputs of the model are conditioned on additional parameters other than the last utterance and the conversation history; may it be a mood (angry, happy, etc...), a set of facts about nearby restaurants (Ghazvininejad et al. 2017), or a set of sentences defining a personality (Li, Galley, Brockett, Spithourakis, et al. 2016, Zhang et al. 2018, Mazare et al. 2018). Unlike unconstrained dialog models, with CDM we can devise simple metrics to determine whether the conditioning was done properly since it is most often akin to the Question Answering task, which has been a popular task in recent years.

Engagingness

Finally, supposing all earlier steps have been solved, a dialog agent will only be of interest if it is able to engage with its dialog partners. By engaging we mean the ability to ask questions and to relate to the dialog partner's statements, to provide with *interesting* informations — as subjective as this notion may be — or even to entertain. Because of this subjectiveness we chose to represent this goal with a dotted line in figure 2.

2 Incremental works

The common drawback of generative models used in dialog tasks is that they tend to default to unappealing answers; "yes/no/I don't know" that make them rather boring to talk to. For ranking answers the fault is generally found in that they lack consistency: if we have in the candidate replies "I hate cats" and "I like cats", both can be picked as most similar to questions likes "Do you like cats?", so that either one of them can be picked and the agent will be found to hold two contradictory opinions. In order to counter these two faults a single solution arises: conditioning the outputs of the agents on a given set of fixed relations it will bear with the world. We will not be talking about the ways to achieve this for generative models and mainly focus on how this is achieved for ranking models.

As is done in Zhang et al. 2018, we take this set of fixed relations with the world as a set of sentences that anchors a personality to the world; "I like cars. My mother works as a nurse. My daughter is 6 y.o.". For a given agent we consider P as set of such sentences and encode them in $\mathbb{R}^d$. This constitutes a memory that we will invoke when confronted to a new context by simply computing the attention on it. Once we have incorporated the memory we need to mix it appropriately.

In this section we expose the work done to try to improve on conditioning as existing in Mazare et al. 2018 which was the starting point of the internship. We provide a review of this article and dive into explored possibilities and inherent limitations of the approach.

2.1 The conditioned Transformer

The aim of Mazare et al. 2018 is to extend Yang et al. 2018 to conditioned dialog models, in the spirit of Zhang et al. 2018. In their work, Yang et al. 2018, use the recently introduced Transformer architecture to train a dialog model based on ranking candidate answers on Reddit data. Hence the encoder structure and the data source were kept and improved to provide conditioning, which was achieved by keeping, for each user, a set of defining sentences. In Mazare et al. 2018 the work on selecting the correct *defining* persona sentences is exposed, with the notable result that adding some noise to these sets helps performance.

A typical training instance is as follow: the model gets an utterance u_C, its successor u_A and the set P of persona sentences of the user responsible for u_A. It then has to retrieve the correct answer that was made, with as many distractors as we have examples in the batch. As described in Yang et al. 2018 we encode u_C and u_A with the transformer architecture (Vaswani et al. 2017). We encode the persona sentences in a simpler way due to their simple structure, with Bag of Words or Deep Averaging Networks (DAN, Iyyer et al. 2015).

$$u_C \longrightarrow v_C \in \mathbb{R}^{d_e} \tag{3}$$

$$u_A \longrightarrow v_A \in \mathbb{R}^{d_e} \tag{4}$$

$$P = \{p_i\}_{0<i\leq m} \longrightarrow M_P \in \mathbb{R}^{m,d_p} \tag{5}$$

Since we cannot expect all persona sentences to be relevant to the current utterance, we have to be able to retain only the relevant information from M_P. This is done via *attention* over M_P, a process that is motivated by Memory Networks (Weston, Chopra, and Bordes 2014). Simply put, we perform the following operation with A a learnable matrix in $\mathbb{R}^{d_e,d_p}$ and σ the softmax function:

$$\alpha_i = \sigma(v_C \cdot A \cdot M_P^T)_i \quad \forall 0 < i \leq m \tag{6}$$

$$m = \sum_{0<i\leq m} \alpha_i M_{P[i,:]} \tag{7}$$

This operation in fact does a weighted sum of the persona sentences, allowing more weight to the most relevant ones, where relevance is obtained via the proxy inner product with the new context. As described in Vaswani et al. 2017 there are many ways to perform attention, for example we could have taken the cosine similarity rather than the inner product in equation 6.

This leaves us with two vectors v_C and m which we now have to combine to obtain a dense representation of the complete context for the answered utterance u_A. The simplest form of combination is to sum these two vectors which is what is done in Mazare et al. 2018:

$$v = v_C + m \tag{8}$$

We then have a vector v that we will rank against a list consisting of encoded answer distractors and the actual encoded answer. The distractors are taken within the batch (the set of examples that will be used to compute a parameter update via the GD optimizer), which ranges in our experiment from 512 to 2048 elements. For our validation we report accuracy in retrieving the correct answer in a batch of 100 elements. In the next subsection we will go over a range of different ranking methods and their results.

$$v \cdot M_{\text{candidates}}^T := r \in \mathbb{R}^K \tag{9}$$

$$loss = \text{CrossEntropy}\big(\sigma(r),\ \text{labels}\big) \tag{10}$$

During training we compute the loss as in equation 10 to compute gradients and update our parameters according to AdaMax (Kingma and Ba 2014), a gradient descent algorithm.

2.2 Exploring ranking methods

As previously described, our models rank candidate answers by order of similarity to the context (and persona). Similarity is usually taken to be the inner product of the encodings, but we can have any function $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ fill the role. We can also devise more intricate ranking methods, which we will cover here.

An obvious candidate ranking function is cosine similarity, i.e. not simply $v \cdot M_{\text{candidates}}^T$ but $\tilde{v} \cdot \tilde{M}_{\text{candidates}}$ with the tilde meaning normalized vectors, i.e. $\tilde{v} = v/||v||_2$. This is often used for a similarity function as it allows to avoid problems where when comparing one vector to two others you might choose the bigger-normed one despite the smaller one being better aligned.

Another way to rank is to consider a key-value store where keys are contexts and values the answers to these contexts. In that setting we can say that our previous method was to rank based on the values. Naturally seeing things this way opens up numerous other possibilities.

Firstly we can rank our context encoding against the keys of our key-value stores and pick the answer as the one made to the most similar context in the candidates.

Qualitatively this method showed improvements on the dialog flow side of dialog, but often fell out topicality wise. A possible interpretation is that our dual encoders are good at detecting implied or lesser topic signals in a question so that topicality is conserved, whereas ranking solely on contexts enables a bigger focus on question forms and dialog flow cues, thereby improving dialog flow.

Secondly we tried to aggregate the information in the top k contexts when ranking on keys. That meant trying to combine the information in a $k \times d$ matrix, so a natural candidate was taking the mean on the first dimension to get a representation in $\mathbb{R}^d$ and re-rank it against contexts to fetch the corresponding answer. This method, that we call "one-hop", did not yield very convincing results despite some initial hope.

A third approach was to use language models to assess the complexity of the candidates. In the same vein as the one-hop method, we establish a short list of k best suited candidates and rerank them by their perplexity, i.e. for a candidate $c = (w_i)_{0<i\leq n_c}$ the value of $\mathrm{PP}_l(c)$ where $\mathrm{PP}_l(c) = \mathbb{P}\left[w_i, 0 < i \leq n\right]^{-1/n}$, the probability of appearance of the sentence normalized by its length. This favors simpler answers, which is arguably a desired behavior for a dialog agent. Although most often this ranking scheme is equivalent to selecting the shortest answer among the top k candidates.

Ranking method	Mean score	Variance	Error
one-hop	2.76	1.01	0.006
simple	**3.32**	1.11	0.004
cosine similarity	2.38	1.30	0.009

Table 2: Scores obtained through a Mechanical Turk experiment totalling 603 dialogues and evaluation by turkers on a 1-5 scale.

In the end ranking as the network was trained proved to be the better evaluated method, as is shown by table 2. These results were obtained through a process that is thoroughly described in section 4. An observation is that cosine under performs, although the method enjoys widespread use. This suggests that norm plays an important role in our encodings.

2.3 Different mixing strategies

In order to improve mixing we tried different strategies more intricated than the summing exposed in equation 8. We mostly have qualitative results to show for these strategies as no real, adapted metric existed at the time. This lack of adapted task meant building new ones, a process exposed in section 4.

The first idea to replace equation 8 was to use a Gated Recurrent Unit (GRU, Cho et al. 2014). The idea being that the GRU will be able to learn to *fully* discard irrelevant persona sentences with its gating mechanisms. The equations for the GRU are recalled below, with $\mathrm{sig} : x \mapsto (1 + e^{-x})^{-1}$ is the sigmoid function.

$$
\begin{aligned}
r_t &= \mathrm{sig}(W_{ir}x_t + b_{ir} + W_{hr}h_{(t-1)} + b_{hr}) \\
z_t &= \mathrm{sig}(W_{iz}x_t + b_{iz} + W_{hz}h_{(t-1)} + b_{hz}) \\
n_t &= \tanh(W_{in}x_t + b_{in} + r_t(W_{hn}h_{(t-1)} + b_{hn})) \\
h_t &= (1 - z_t)n_t + z_t h_{(t-1)}
\end{aligned}
$$

The first two equations describe the gating variables, r_t being the reset gate and z_t the update one. Our hope is that the W_*, b_* parameters will be learnt to be able to discard

persona sentences irrelevant to the context, denoted x in these equations. We can use the GRU in two ways; either treating the set of encoded persona as a "sentence", which adds a sequentiality dimension to the problem, superfluous in our case since the ordering of the persona sentences has never been of importance, or using it after the Memory Network to have equation 8 become $v = \text{GRU}(v_C, m)$ to learn to discard persona information altogether.

Another idea was to offer a possibility to attention to default to an "empty" persona sentence so as to allow not to condition the output. This was motivated by the fact that, in dialog, identities do not motivate all utterances, e.g. when asking about the weather, we need not mobilize knowledge involving our tastes or other personality defining traits. Practically this was done by creating a new token in the vocabulary and appending to all persona sets a sentence consisting of this sole `<PER>` token.

We experimented with simple changes at inference time (i.e. not during training), firstly with weighted sum, i.e. instead of $v + m$ do $\alpha v + \beta m$. For the sake of simplicity we experimented with $\alpha = 1, \beta \in [0, 1]$, although this parameter could also be learned in some way, e.g. either via a vector C performing $v + (m \cdot C^T) * m$ or earlier with a matrix C at step 6. However we did not experiment with the learning of this parameter.

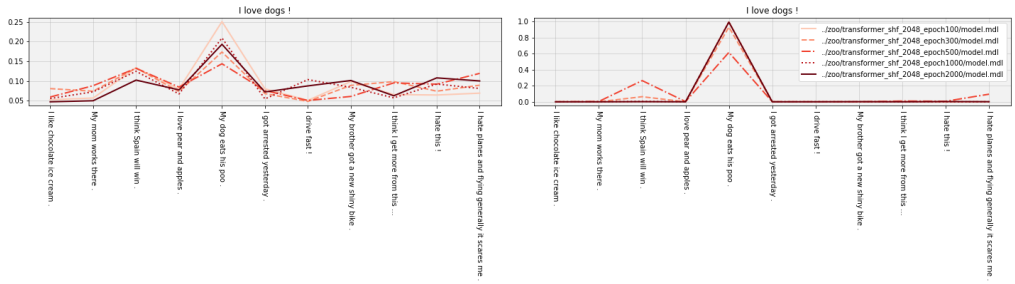

Figure 4: Attention with $\tau = 1$ (*left*) and $\tau = 0.1$ (*right*)

The second change is τ, the temperature of the softmax function, which is initially set to 1 (recall equation 6). In Vaswani et al. 2017 the authors report good performances for $\tau = \sqrt{d}$ with d the dimensionality of the encodings. We performed experiments for different taus at inference time only, see figure 4 for an overview of its effects for a range of different models.

Also experimented was something we dubbed "dropsample", which is performing $v + x * m$ where x is a random variable following the Bernoulli law with parameter p, $x \sim \mathcal{B}(p)$. This is motivated by an unpublished experiment lead by Samuel Humeau on conditioning dialog models with moods as categorical variables. This would serve as a regularization method, and it proved beneficial in the mood conditioning experiment.

2.4 Limitations of ranking based agents & conversation history

Limitations of ranking based agents for CDM

The following setting illustrates the inherent problem with conditioning on dialog outputs using fixed candidates: if we have in the user persona the sentence "The XYZ movie is my favorite horror movie." and ask them "What's your favorite horror movie?", if in the set of candidates we don't have any sentence mentioning the XYZ movie, we will never have a chance at answering correctly. More formally, if we want our conditioning to be effective we need to have the persona sentences being about a finite set of items I so that we can find in P any sentence relating to any item of I in every way possible - possibly partitioned into positive/negative statements.

Keeping ranking methods, an improvement could be to combine candidates along different criteria and order them with a LSTM. However this combination problem is a hard one and non smooth one, implying the use of either hard coded choices, or of reinforcement learning methods to keep generality and robustness. Partitioning the dialog problem into the dialog flow and the topicality problem as exposed in 1.3, we could choose candidates answering these two issues, i.e. a short first candidate for dialog flow, and a second one for topicality. Of course this doesn't solve the consistency problem we are trying to solve with conditioning, and possibly even adds to the problem since any part of the arrangement have to be consistent between them.

Our opinion is that this could be an interesting line of questioning for future works given the successes we have with ranking based algorithms.

Integrating conversation history

In Mazare et al. 2018 the dialog agents do not incorporate dialog history when choosing candidates. We have found it not to be noticeable when talking with agents as if with a child, i.e. by repeating contexts in each utterance and limiting ambiguities or implicits. However we would want the bot to be able to infer these information from the previous utterances. In a sense we are finding ourselves with a problem similar to the one exposed in subsection 2.3, i.e. one of mixing information from one source with one from the other.

Our solution was to add a new token in our dictionary, the `<SOC>` token denoting "start of comment". Each input to the model has to be prepended such a token, and depending on how many past utterance we want to condition on we will prepend them as well with this token as separation. This works well enough, as we can see in table 3.

Model	Mean score	Variance	Error
Transformer without history	2.91	1.12	0.11
Transformer with 3 last utterances	3.35	1.17	0.15

Table 3: Scores obtained through a Mechanical Turk experiment totaling 603 dialogs and evaluation by turkers on a 1-5 scale.

We could have added it to the persona sentences and trust the Memory Network to use them when needed, but the intuition behind this choice is that discussion history doesn't play the same role as the persona. This asks the question of incorporating the persona sentences in the string used as input with some other delimiter. We did not make experiments to try this, although it would be interesting to measure how the Transformer would handle several hundreds of tokens long inputs. Our opinion is that it would also play badly conditioning-wise with all the persona sentences being added at the same time, the noise would be too strong and the Transformer doesn't have a way to choose which one would be more relevant to the context.

3 Original works: batching

We've exposed earlier how our model is trained, which is by correctly guessing the actual answer made to a given context among a given number of candidates. These other candidates we call distractors can be chosen according to a given policy and in a given number. While we initially focused on their number and the consequences it would have on our accuracy, we realized that making changes in the distribution of the candidates yielded big improvements.

In this section we will provide some refreshers on negative sampling and its uses, then overview the structure of our dataset, the difficulties of characterizing distributions on utterances and the end effect and empirical conclusions of careful negative sampling.

3.1 Noise contrastive estimation & negative sampling

Noise contrastive estimation (NCE, Gutmann and Hyvarinen 2010) is a general parameter estimation technique relying on nonlinear logistic regression to discriminate between noise and observed data. The NCE is an unbiased estimator that was developed to avoid costly computations in language models where we frequently sum probabilities over whole vocabularies that are at the very least in the tens of thousands words big.

In Mikolov, Sutskever, et al. 2013 the authors introduce the Negative Sampling algorithm with the same motivation, which derives from NCE. Negative sampling is a subclass of NCE as shown by Dyer 2014 which is easier to use as it doesn't require the specification of the noise distribution and only needs samples. However doing this we lose some information on the values of the probabilities we are trying to approximate, which is why negative sampling is best suited for the learning of representation rather than language models.

3.2 The Reddit dataset

Reddit is a forum with a galaxy of "subreddits" that are dedicated to specific interests. Users can submit arbitrary links from the Internet that are meant to be commented by the community. At its core it is a tree, with level 1 nodes being subreddits, that all have a certain number of posts (level 2 nodes), that each have discussions attached. Any discussion starts with a root comment that is then replied to any given number of times, forming another subtree.

The source of our data was a torrent file whose link is now unavailable, but the data is still query-able[4][5][6] for replication. Our dataset comprises an approximate 900Go which has been evenly divided into 1,000 chunks by hashing the string uniquely identifying posts, each chunk has on average 1,600,000 unique comments. A dictionary is compiled consisting of 684,459 unique words appearing in the dataset, while in our experiment we keep the first 250,000 most frequent ones.

For our model as described in 2.1 it takes an approximate 72 hours on 8 GP100 GPUs to train a full epoch, which is for us chunk 0 to 998, leaving chunk 999 for validation purposes. Since the chunks are made after iterating through the whole dataset and computing post by post its hash value, the chunks are sequences of reddit posts' content, i.e. we will get all the comments and replies to one post in a subreddit A, then all content related to a post that belongs in another subreddit B, etc... And since the hashing function does its

[4] https://www.reddit.com/r/bigquery/comments/3cej2b/17_billion_reddit_comments_loaded_on_bigquery/

[5] https://www.reddit.com/r/bigquery/comments/5z957b/more_than_3_billion_reddit_comments_loaded_on/

[6] http://files.pushshift.io/reddit/comments/

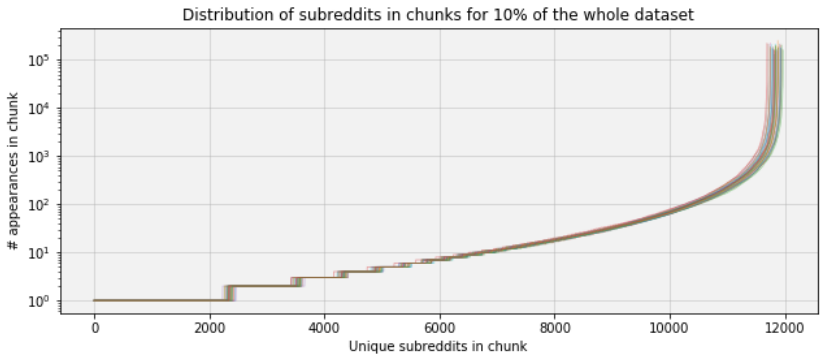

Figure 5: Subreddit distributions in chunks

job reasonably well, the chunks are evenly distributed in terms of subreddit frequency as is attested by figure 5.

When training our models we shuffle the chunks, so that a batch is composed of a given number of context-answer pairs that are sampled randomly. While this seems like a benign operation, we have come to the conclusion that it in fact bears a lot of consequences. To understand the effects of this let's imagine a batch of size 3 like the following:

I think the Boeing 747 is the best.	→	I don't know the A380 is cool too!
Did you go sailing in the Bay?	→	No! Only practiced on the East Coast...
The actors are soo bad in the sequel lol	→	The soundtrack was dope tho

Table 4: A small batch of size 3

To accurately predict the answer for each context, you do not to model any form of dialog flow, modeling topicality is more than enough. While this example is caricatural, we have found that it is what happens in the shuffled batches, as also attested by figures 15 and 14. Table 5 details this intuition for batches of size 512 and 2048, where we take subreddits as a proxy for topicality, an claim detailed in the next subsection.

Batch size	Composition	Expectation of number of same subreddit comments in batch
512	Shuffled	10.2 (1.99%)
	Non-shuffled	107.1 (20.92%)
2048	Shuffled	37.7 (1.84%)
	Non-shuffled	285.7 (13.95%)

Table 5: A small batch of size 3

3.3 Shuffling or not, introspecting distribution on sentences and batch composition

We've exposed what happens when we shuffle chunks in the previous section and the effect we understand it has on our representations. Based on this, and following the dialog permutation task exposed in Ritter, Cherry, and Dolan 2010, we devised a way to disrupt this topicality signal preponderance by simply *not* shuffling the chunks. Doing so, one finds in a batch elements from the same discussion, thereby making the task harder since topicality is, at least locally, preserved. This forces the representations to account for other dialogical cues, as exposed in section 1.3.

Our best proof for these claims is the superiority of models that are, all other things being equal, trained on non-shuffled chunks on human evaluation — a process described in section 4.1, with scores exposed in tables 6 and 7.

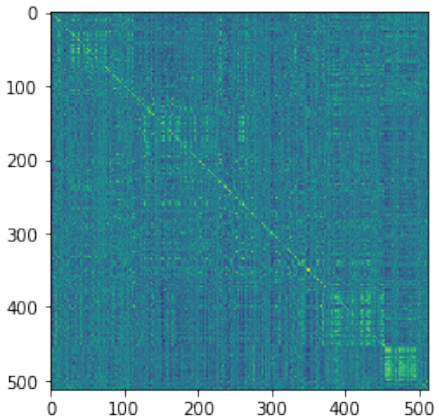

Figure 6: A context-answer similarities matrix for a non-shuffled batch. Similarities obtained by inner product of sentences encoded by a trained shuffled Transformer.

The preceding claims point towards an important role of the choice of the negative samples and their consequences on representations and dialog abilities. This is further supported by a successful fix we implemented to prevent our agents to repeat their utterances multiples times (a phenomenon we dubbed the parrot effect), which consisted in adding an element of the dialog history in the negative samples. This switch of shuffling or not our chunks is quite simplistic, and had our dataset been shuffled initially it might not have come to mind to keep the structures inside the batches. We can look at figure 6 to understand what happens in a non-shuffled batches (and at figures 14 and 15 for more evidence): the encoders can retrieve topicality blobs, which correspond to reddit post. This suggests that our encoder are topicality efficient, and that our non-shuffled batches effectively have some regularity topicality-wise. This poses the question of how to precisely inform the composition of our batches.

This reveals uneasy due to the nature of our space; in Word2Vec the authors inform their negative sampling with a flattened unigram distribution (frequency of words raised to 3/4). We do not have a simple way to describe a natural probability of the space of all possible grammatical sentences — although one could object that with a strong language model one could compute the probability of each comment and normalize it across the whole dataset to provide with a similar distribution as the unigram. However this is not desirable firstly because computing the probability of each comment for a dataset as big as reddit is not reasonable, then because normalizing it afterwards would probably go beyond our floating point capabilities, also because sampling from this distribution would also be too costly. Furthermore this would not provide with hard negative samples since language models do not care about topicality, implying that batches would be rather composed with lots of simple sentences (short and common vocabularies) and few more exotic ones. In order to provide with a better batch composition scheme an idea could be to partition the dataset by topics, so as to have some control over topicality and to be able to make experiments with 10, 20, 30, ...100% of the batch in the same topic and observe the consequences on dialogical abilities.

We've experimented with a reordering of the chunks by subreddits, under the assumption that given reddit's functioning they'd be satisfying proxies for topicality. This

assumption is arguably only true for moderate sized subreddits, since bigger ones tend to be very eclectic in nature: "r/pics" is one of the bigger subreddits and the posts can vary wildly in topics depending on what's in the pictures, same goes for "r/askreddit", the biggest one with over 180 million comments. We only selected subreddits with more than 50,000 comments and less than 2 millions, which yielded mostly the subreddits dedicated to hobbies and passions. Doing so results in batches where the totality of the elements are linked to a single topic, making the task much harder than the initial one. The resulting models are not very good at dialog, and not even at dialog flow, and rather seem to be very focused on keywords. Unfortunately we don't have scores on DialogEval to support these statements.

Rather than focusing on topicality for batch composition we can also look at similar works. Recently Bose, Ling, and Cao 2018 have suggested training reinforcement learning algorithms to dynamically choose elements for negative sampling. There is also some evidence on hard negative mining (Schroff, Kalenichenko, and Philbin 2015) in faces recognition or image problems, where researchers only keep hard examples, i.e. ones were the model's classification is wrong or done with little confidence. We could also tie the notion of curriculum learning (Bengio et al. 2009) which would mean training on random distribution first, due to the easier nature of the setting, and then gradually increase the proportion of same topic negative samples.

4 Original works: Metrics, DialogEval & new datasets

Whenever working with language one soon hits the metric problem, in that metrics don't exactly correspond to our goals, a case we found ourselves in when trying to compare models developed in subsection 2.3. More generally for the dialog task we find ourselves lacking metrics that correlate well with human judgment. In the case of our ranking based dialog models, the metric is the one used at training: hits at k for K candidates, meaning is the answer in the first k most similar candidates to the context. As we have seen in the previous section the choice of distractors is of importance and tells us a lot about our model.

In this section we will look at some proposed metrics for dialog that should correlate with human evaluation, the setting in which these human evaluation were gathered, and present our DialogEval toolkit gathering these metrics for fast and easy benchmarking of dialog models. We also offer some insights on metrics that did not correlate well, particularly Natural Language Inference (NLI)

4.1 Gathering human evaluations

In order for our DialogEval set of tasks to be interesting they have to be correlated with human judgment, hence our need to provide such data. Following works done in Zhang et al. 2018, we used Amazon's Mechanical Turk[7] platform to perform the human evaluations. We present here the task that was submitted to the Mechanical Turkers and from which derive the human scores presented earlier.

Our base code derives from the ConvAI2 code[8] which is itself based on the ParlAI framework (Miller et al. 2017). It features a simple discussion task we dubbed "Have a break, have a chit-chat!", where the Turker is greeted by instructions to have a light discussion with other agents that might be human or chatbots, see figure 7 for an example start of the task.

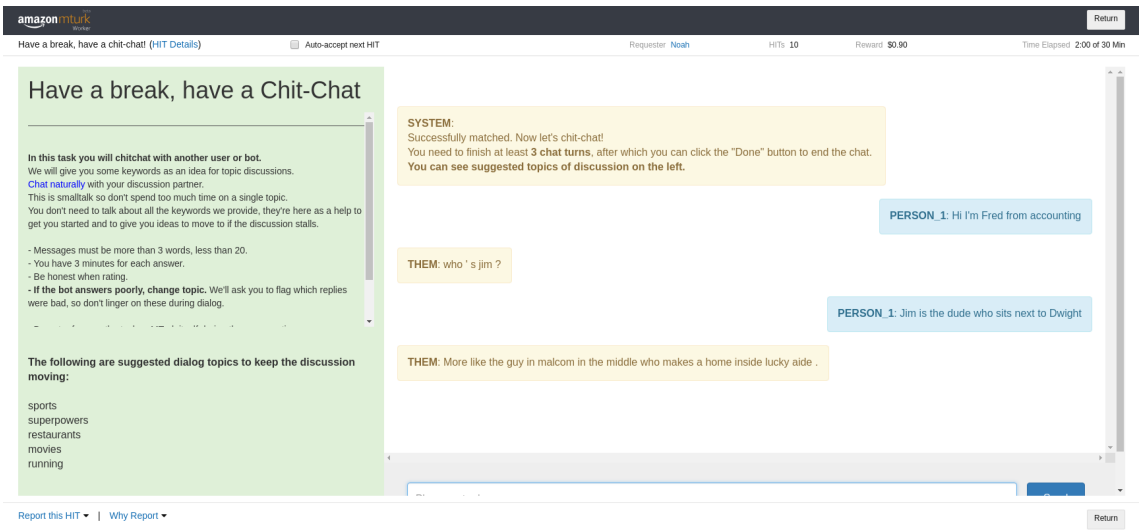

Figure 7: First rounds of dialogs in our Mechanical Turk task

We instruct turkers to have a simple discussion with the agent, to change topic regularly and to ignore bad outputs if any. This was necessary because in preliminary experiments we saw turkers trying to get our agents to acknowledge their errors on 4 to 5 successive dialog rounds, yielding poor conversations and evaluations. Upon completion (6 rounds)

[7] https://www.mturk.com/
[8] http://convai.io/

we ask the turkers to rate the dialog they've had on a scale from 1 to 5, with 5 being human, then we outlay the conversation and ask them to tell us what dialog rounds were bad. Finally we ask them if they'd want to talk to the bot again, a proxy for the engagingness of the model, as done in Venkatesh et al. 2018. We expose the evaluation procedure in figures 11, 12 and 13.

At first we asked more questions, particularly we asked first to select rounds that were bad, then rounds that were not topical. However this gave poor results since we didn't properly define topicality beforehand, which revealed too long a process to be included in the task. Hence this somewhat simplistic task, but that we think is a strong baseline for human evaluation given that our discussions are diverse in topic and dynamics, the set of instruction works since our turkers have the desired behavior (ignore bad outputs, flag them later during evaluation) and standard deviation is reasonable at 1 point out of 5.

We launched two preliminary versions to test how the turkers would react and made these decisions on this basis. Our first run included models trained on shuffled and non shuffled reddit chunks, different batch sizes and different ranking schemes, its results are exposed in table 6. Its aim was to confirm our intuition on batch composition and to give us insights on ranking methods.

Model	Ranker	Score	Std	Error
transformer_noshuffle_2048_context_as_negative-3h	simple	3.67	0.96	0.135
transformer_noshuffle-3hist	simple	3.39	1.17	0.148
transformer_halfshuffled	simple	3.3	1.01	0.147
transformer_noshuffle	simple	3.29	1.05	0.156
transformer_noshuffle_2048_context_as_negative-3h	simple_normalized	3.28	1.01	0.137
transformer_vanilla	simple	3.03	1.17	0.144
transformer_noshuffle_2048_context_as_negative-3h	onehop	2.89	0.98	0.148
transformer_vanilla	onehop	2.84	1.02	0.154
transformer_halfshuffled	onehop	2.7	1.1	0.162
transformer_halfshuffled	simple_normalized	2.69	1.11	0.156
transformer_noshuffle	onehop	2.63	0.89	0.132
transformer_noshuffle	simple_normalized	1	0	0

Table 6: The first round of human evaluations, totaling 650 conversations.

In table 7 we report results for the second evaluation campaign, which had humans talking to humans, a fixed number of dialog rounds (before that turkers could continue the discussion after a minimum number of turns, however they never did so), and a more diverse range of models. We observe a lower mean than for the first campaign, which we attribute to the presence of humans, which raises the standard for "good" conversation. We included a generative model, the baseline for the ConvAI2 challenge mentioned earlier which is based on a Seq2Seq architecture, and include a ranker with a simple Bag of Words encoders. We included our best model as per the first experiment at different moment during training time for two reasons: one as it enables us to detect training dynamics with respect to our validation metrics, and secondly as it increases our number of different models to get better correlations.

4.2 Extending ranking tasks

Extending ranking metrics was done in two ways; first of all by looking into the distribution D underlying the candidates, and secondly in realizing that there was more to ranking than just precision at k for K.

Model	Score	Std	Error
humans	5	0	0
transformer_nos_2048_3hist_histneg_epoch2000	3.106	1.096	0.16
transformer_shf_2048_epoch1000	3.002	1.164	0.148
transformer_shf_2048_epoch500	3.00	1.047	0.161
transformer_nos_2048_3hist_histneg_epoch300	2.976	1.047	0.164
transformer_nos_2048_3hist_histneg_epoch1000	2.938	0.944	0.136
transformer_shf_2048_epoch2000	2.87	0.969	0.143
transformer_shf_2048_epoch300	2.812	1.088	0.136
transformer_nos_2048_3hist_histneg_epoch500	2.646	1.09	0.157
transformer_shf_2048_epoch100	2.529	0.957	0.134
transformer_nos_2048_3hist_histneg_epoch100	2.284	1.034	0.126
convai_s2s	2.093	1.007	0.154
bow-noshuffle-2048-3hist	1.479	0.736	0.106

Table 7: The second round of human evaluations, totaling 603 conversations.

The Dialog Permutation task

Our initial metrics were hits@k/K for candidates sampled randomly in a validation chunk of the Reddit data. The proposed way to describe ranking metrics is hits@k/K/D where D denotes the underlying distribution or source of the data. Developing an idea present in Ritter, Cherry, and Dolan 2010 we introduce the dialog permutation task, for which we provide a sample in table 8.

Keeping the dichotomy between topicality and dialog flow, our intent with the dialog permutation task is to provide a way to assess the ability of the models to capture the dialog flow information. This is partly done under the assumption that in a discussion the topic is, at least locally, preserved, so that topical cues can not help the model choose between candidates. The sources for these dialogs are taken in the Ubuntu dataset (Lowe et al. 2015), the PersonaChat dataset (Zhang et al. 2018), and the Cornell Movie Dialog Corpus (Danescu-Niculescu-Mizil and Lee 2011). Each of these have defining characteristics, with the Ubuntu dataset being technical (although we filtered the conversations so as to maximize vocabulary overlap with our more mainstream one from Reddit), the PersonaChat one being volatile, with dialog partners changing topics at a high rate (one or two rounds only), and the Cornell Movie Dialog Corpus one being perhaps closer to standard chit-chat, albeit theatrical by nature and where dialog partners share a lot of knowledge exterior to the conversation.

Context	Candidates
When's school over ?	— He doesn't want to spend the weekends with you anymore. — I ' ll be by Saturday to pick him up. — Friday... God, I hate the thought of him home all day.
I'm back in Grand Rapids .	— I've got a meeting... — In G.R.? Why? — Can you meet me in about an hour? At the Pantlind Hotel?

Table 8: Dialog permutation samples, sourced from the Cornell Movie Dialogs corpus (Danescu-Niculescu-Mizil and Lee 2011)

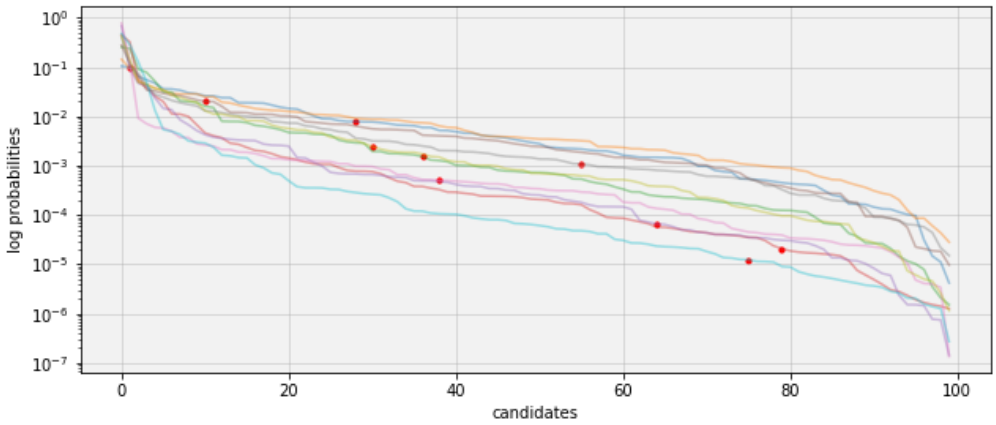

Figure 8: Log probabilities for 5 different contexts with the Transformer model, the red dots signaling the position of the correct answer.

Diversifying ranking metrics

When ranking K candidates one can look at different metrics; the one we started with was the accuracy in the first k choices. Typically we look at P@1/100, P@10/100 at validation, while at inference we rank millions of candidates. A good metric is the mean rank of the actual candidate, for example for two performances at 50% on P@10/100, you can find one with mean rank $(1 + 11)/2 = 6$ and one with mean rank $(10 + 50) = 30$, making for two very different models. Since we rank by assigning probabilities to each candidates, it is of interest to look at the difference of probabilities between the selected candidate and the actual one. For that purpose we look at confidence as defined in equation 11 where $\mathbb{P}_i$ is the current distribution over the candidates $\{c_i\}_{0<i\leq K}$ and the correct candidate is noted c_i^*.

$$\text{Conf}_i = \max_{0<j\leq K} \mathbb{P}_i\left[c_j\right] - \mathbb{P}_i\left[c_i^*\right] \tag{11}$$

The idea behind this is to characterize the error that was made; if the confidence is *low*, it means, whatever the position of the actual candidate, that the candidates placed before are somewhat similar, i.e. the error is perhaps not too big. However a *bigger* confidence means the model got something wrong and placed first a candidate that shares little similarity with the correct one. We can look at the rank lines in figure 8 to visualize these effects.

Another idea is to use a standard metric in search engines: Discounted Cumulative Gain (DCG, Järvelin and Kekäläinen 2002), which in our setting would be expressed in the following equation, assuming we have ordered the c_i by their probabilities:

$$\text{DCG} = \sum_{0<i\leq \text{rank}(c^*)} \frac{\mathbb{P}[c_i]}{\log i} \tag{12}$$

4.3 The DialogEval toolkit

The motivation for the building of a code repository enabling simple benchmark computations takes root with the SentEval (Conneau and Kiela 2018) and GLUE (Wang et al. 2018) initiatives for sentence representations evaluation. These two initiative respond to similar challenge as ours: evaluating sentence representations is not straightforward and the best way to do so is to look at their performances in different downstream tasks. Our intent is therefore to provide the dialog modeling community with a similar set of tasks

and code to fast track evaluation, which we aim to do with the DialogEval repository that will be released in the near future. This reveals necessary since dialog metrics are hard to come by, as attested by Liu et al. 2016, an attempt to evaluate dialog models with translation metrics that are shown not to correlate well with human judgment. The recent papers Guo et al. 2018, Venkatesh et al. 2018 are offering an interesting first attempt, but their topic classifier is not released for public use, hence limiting its impact.

With the previous subsection we have given ourselves a way to assess the ability of a model to capture dialog flow, and by selecting candidates randomly we rather assess the ability of the model to capture topicality — although this could be done with a topic classifier such as done in Guo et al. 2018, which gives us a good coverage of our pyramid of language skills such as presented in figure 2. We present later in this section tasks to evaluate consistency as implemented by persona-conditioned dialog models.

DialogEval will only be appealing to the community if the tasks it offers are shown to correlate well with human judgment, a perspective that motivated the human evaluation process presented in section 4.1. In table 11 we report correlation values for a range of different tasks, with the following metrics:

- "acc": accuracy, i.e. P@k/K
- "rank": mean rank
- "uconf": the confidence as described in equation 11
- "nconf": the confidence as described in equation 11, normalized by the rank

However these values can only serve as indications since our number of models is too small to definitely confirm any result. It is intended as future work which will be done in the near future.

In order to be thorough we would also need to check for consistency since we presented earlier (section 2.3) attempts at refining conditioning. We present here two tasks to check the ability of the model to condition on persona sentences, which is very close to a Question Answering task.

Chocolate

Persona	Questions	Answers
I don't like Snape.	Do you like Snape?	Yes!
I despise the abominations.	Snape is so awesome.	Yes.
I hate that word now.	Do you love Snape too?	Yeah it's great.
I don't like spiders.	Snape is the best IMO , do you agree?	I completely agree with you on this!
I hate those people.	Do you hate Snape?	True.
I hate his nuts.	Snape is really bad.	No!
I like this video alot.	Do you despise Snape too?	No.
I love art and books.	Snape is the worst IMO , do you agree?	Definitely not.
I love Paul.		False.
I love Hal!		No way this is bullshit.
I love this picture.		I have to disagree firmly on this!

Table 9: The Chocolate dataset: an instance with 10 distractors in the persona, sampled from similar polarized sentences.

The Chocolate dataset, with an example in table 9, aims to check the ability of the model to effectively condition its outputs based on a set of defining traits. Particularly

with Chocolate we want to see if the model can retain and reason on polarized relations: if the agent *loves* chocolate, when asked "Do you like chocolate?" it should choose among the possible answers one that is positive. We also included assertions in the questions to evaluate agreement performance. We use distracting persona sentences to make the task resemble our actual setup at inference time and evaluate how the agent chooses the relevant items in memory. The choice of the underlying distribution behind these distracting persona sentences was made to be sentences similar to the main one we will question, i.e. polarized, short sentences in the form of "`I <verb> <object>`" where `<verb>` is like/love/hate/... or simply the persona sets of the user who made these comments on Reddit, as devised by Mazare et al. 2018.

It turns out that this task is quite hard, with our best models performing only slightly above random guess (50%) at 61.7%. We explain this difficulty by the absence of word overlaps between question, persona and answers, which would require some kind of reasoning or state tracking to be solved effectively.

PersonalQA

Persona	Question	Answers
i entered the world with some issues.	what animal do i like?	Capra aegagrus
i mow the lawn.		homogenized milk
i am an immigrant.		Lolita
i have trouble with food when i do not have anything to do.		Renelle
i have pets.		Maybelle
i am employed in the education industry.		Cara
i enjoy eating pastries.		bootleg
the person that sings bad blood is my most beloved musician.		Florentia
i sell people their backup plans.		devil's food cake
i have very expensive taste in vehicles.		oxheart cherry
Capra aegagrus is so cute.		...

Table 10: The PersonalQA dataset: an instance with 10 distractors in the persona, sampled from the PersonaChat dataset.

The PersonalQA (PQA) dataset is similar to the Chocolate one in aim, but assesses rather the ability to answer the question "What's your favorite ice cream flavor", i.e. one solution in many candidates instead of simple yes or no questions. As seen in table 10 we also use distracting persona sentences, sampled from the PersonaChat persona sentences.

Interestingly, we first thought the PQA task would be harder to solve than the Chocolate one due to the questions being posed, quite open for the first one ("What") and simple yes/no questions for the latter. In fact this is not true, our best models perform as high as 70%, for random guesses being below 10% depending on the number of distracting answers we take, making this task easier to solve than Chocolate. Our interpretation is the presence of word overlaps between persona, question and answers which help guide the model to the right answer.

4.4 Natural Language Inference

Working our way into good metrics we imagined that the Natural Language Inference (NLI) task would be a good one. To be able to do dialog well we would need the model to be able to tell apart contradiction and entailment as they're inherent parts of our use of dialogs. We used the Stanford NLI dataset (SNLI, Bowman et al. 2015) which is standard for this task.

It turns out that it is a very bad metric that doesn't correlate with dialog — although this claim lies on a limited number of models, so that we will only claim it for our Transformer model, as we can see with figure 9. Our interpretation is that the dialog and NLI tasks induce different geometries for the representations, in that for NLI we want our representations of contradicting sentences to be far apart, where for dialog we would want them to be close one to the other as dialog is often the exchange of contradicting opinions. With figure 10 we look at scores during training, which reveals that we get better before getting worse at SNLI, while human evaluation rises with the epochs.

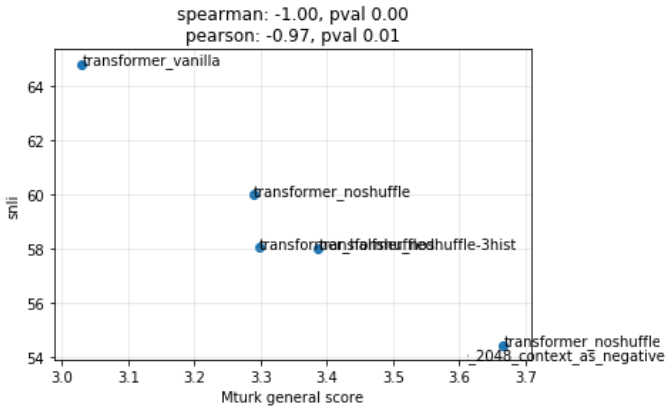

Figure 9: SNLI correlations

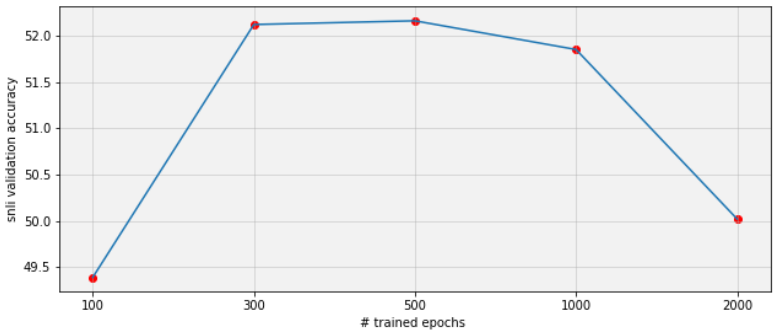

Figure 10: SNLI correlations during training

Conclusion

The dialog task is far from being solved, however we believe some of our contributions will help build better dialog models.

Our conceptual framework decomposing dialog into individual skills should help understand the dialog task, to help detect and attend to the most common modes of failure of dialog models. The DialogEval toolkit, based on this framework, should also help foster the development of new algorithms by providing reliable ways to compare dialog models between them without having to rely on expensive human evaluations, or at least to serve as a first approximation of the model's abilities.

Our considerations on batch composition could be the start of new works, and it is our belief that smart batch composition could yield significant improvements. It is still unclear however what form these improvements would take.

Considering the conditioned dialog models, our intuition is that the best way forward lies somewhere between generative and ranking methods, whether it be the assembling of different pre-conceived candidates, or by modifying candidates with generative methods (Weston, Dinan, and Miller 2018).

The works presented here span multiple problematics and lay the foundations of future works, both in the evaluation process which needs proper correlation numbers to be accepted by the scientific community and in the conditioned dialog setting.

References

Arora, Sanjeev, Yingyu Liang, and Tengyu Ma (2017). "A SIMPLE BUT TOUGH-TO-BEAT BASELINE FOR SEN- TENCE EMBEDDINGS". In: p. 16.

Bengio, Yoshua et al. (2009). "Curriculum learning". In: *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*. the 26th Annual International Conference. Montreal, Quebec, Canada: ACM Press, pp. 1–8. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553380.

Bojanowski, Piotr et al. (July 15, 2016). "Enriching Word Vectors with Subword Information". In: *arXiv:1607.04606 [cs]*. arXiv: 1607.04606.

Bose, Avishek Joey, Huan Ling, and Yanshuai Cao (May 9, 2018). "Adversarial Contrastive Estimation". In: *arXiv:1805.03642 [cs]*. arXiv: 1805.03642.

Bowman, Samuel R. et al. (2015). "A large annotated corpus for learning natural language inference". In: Association for Computational Linguistics, pp. 632–642. DOI: 10.18653/v1/D15-1075.

Cho, Kyunghyun et al. (June 3, 2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *arXiv:1406.1078 [cs, stat]*. arXiv: 1406.1078.

Conneau, Alexis and Douwe Kiela (Mar. 14, 2018). "SentEval: An Evaluation Toolkit for Universal Sentence Representations". In: *arXiv:1803.05449 [cs]*. arXiv: 1803.05449.

Danescu-Niculescu-Mizil, Cristian and Lillian Lee (June 15, 2011). "Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs". In: *arXiv:1106.3077 [physics]*. arXiv: 1106.3077.

Davidson, Donald (Dec. 1982). "Rational Animals". In: *dialectica* 36.4, pp. 317–327. ISSN: 0012-2017, 1746-8361. DOI: 10.1111/j.1746-8361.1982.tb01546.x.

Du Bois, John W. (Jan. 1, 2014). "Towards a dialogic syntax". In: *Cognitive Linguistics* 25.3. ISSN: 1613-3641, 0936-5907. DOI: 10.1515/cog-2014-0024.

Dyer, Chris (Oct. 30, 2014). "Notes on Noise Contrastive Estimation and Negative Sampling". In: *arXiv:1410.8251 [cs]*. arXiv: 1410.8251.

Firth, John R (1957). "A synopsis of linguistic theory, 1930-1955". In: *Studies in linguistic analysis*.

Ghazvininejad, Marjan et al. (Feb. 7, 2017). "A Knowledge-Grounded Neural Conversation Model". In: *arXiv:1702.01932 [cs]*. arXiv: 1702.01932.

Grice, H Paul (1975). "Logic and conversation". In: *1975*, pp. 41–58.

Gundel, Jeanette K. (Feb. 1985). "'Shared knowledge' and topicality". In: *Journal of Pragmatics* 9.1, pp. 83–107. ISSN: 03782166. DOI: 10.1016/0378-2166(85)90049-9.

Guo, Fenfei et al. (Jan. 10, 2018). "Topic-based Evaluation for Conversational Bots". In: *arXiv:1801.03622 [cs]*. arXiv: 1801.03622.

Gutmann, Michael and Aapo Hyvarinen (2010). "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models". In: p. 8.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory". In: *Neural Comput.* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735.

Iyyer, Mohit et al. (2015). "Deep Unordered Composition Rivals Syntactic Methods for Text Classification". In: Association for Computational Linguistics, pp. 1681–1691. DOI: 10.3115/v1/P15-1162.

Järvelin, Kalervo and Jaana Kekäläinen (Oct. 1, 2002). "Cumulated gain-based evaluation of IR techniques". In: *ACM Transactions on Information Systems* 20.4, pp. 422–446. ISSN: 10468188. DOI: 10.1145/582415.582418.

Kingma, Diederik P. and Jimmy Ba (Dec. 22, 2014). "Adam: A Method for Stochastic Optimization". In: *arXiv:1412.6980 [cs]*. arXiv: 1412.6980.

Li, Jiwei, Michel Galley, Chris Brockett, Jianfeng Gao, et al. (Oct. 11, 2015). "A Diversity-Promoting Objective Function for Neural Conversation Models". In: *arXiv:1510.03055 [cs]*. arXiv: 1510.03055.

Li, Jiwei, Michel Galley, Chris Brockett, Georgios P. Spithourakis, et al. (Mar. 19, 2016). "A Persona-Based Neural Conversation Model". In: *arXiv:1603.06155 [cs]*. arXiv: 1603.06155.

Liu, Chia-Wei et al. (Mar. 25, 2016). "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation". In: *arXiv:1603.08023 [cs]*. arXiv: 1603.08023.

Lowe, Ryan et al. (June 29, 2015). "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems". In: *arXiv:1506.08909 [cs]*. arXiv: 1506.08909.

Mazare, Pierre-Emmanuel et al. (Aug. 2018). "Training Millions of Personalized Dialogue Agents". In: EMNLP, p. 5.

Mikolov, Tomas, Kai Chen, et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Ilya Sutskever, et al. (2013). "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*, pp. 3111–3119.

Miller, Alexander H. et al. (May 18, 2017). "ParlAI: A Dialog Research Software Platform". In: *arXiv:1705.06476 [cs]*. arXiv: 1705.06476.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Ritter, Alan, Colin Cherry, and Bill Dolan (2010). "Unsupervised Modeling of Twitter Conversations". In: *ACL*, p. 9.

Salton, Gerard and Christopher Buckley (1988). "Term-weighting approaches in automatic text retrieval". In: *Information processing & management* 24.5, pp. 513–523.

Schroff, Florian, Dmitry Kalenichenko, and James Philbin (June 2015). "FaceNet: A Unified Embedding for Face Recognition and Clustering". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682. arXiv: 1503.03832.

Searle, John R and John Rogers Searle (1969). *Speech acts: An essay in the philosophy of language*. Vol. 626. Cambridge university press.

Sordoni, Alessandro et al. (June 22, 2015). "A Neural Network Approach to Context-Sensitive Generation of Conversational Responses". In: *arXiv:1506.06714 [cs]*. arXiv: 1506.06714.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Cambridge, MA, USA: MIT Press, pp. 3104–3112.

Vaswani, Ashish et al. (June 12, 2017). "Attention Is All You Need". In: *arXiv:1706.03762 [cs]*. arXiv: 1706.03762.

Venkatesh, Anu et al. (Jan. 10, 2018). "On Evaluating and Comparing Conversational Agents". In: *arXiv:1801.03625 [cs]*. arXiv: 1801.03625.

Wang, Alex et al. (Apr. 20, 2018). "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: *arXiv:1804.07461 [cs]*. arXiv: 1804.07461.

Weizenbaum, Joseph (1966). "ELIZA—a computer program for the study of natural language communication between man and machine". In: *Communications of the ACM* 9.1, pp. 36–45.

Weston, Jason, Sumit Chopra, and Antoine Bordes (Oct. 14, 2014). "Memory Networks". In: *arXiv:1410.3916 [cs, stat]*. arXiv: 1410.3916.

Weston, Jason, Emily Dinan, and Alexander H. Miller (Aug. 14, 2018). "Retrieve and Refine: Improved Sequence Generation Models For Dialogue". In: *arXiv:1808.04776 [cs]*. arXiv: 1808.04776.

Winograd, Terry (1971). *Procedures as a representation for data in a computer program for understanding natural language*. Tech. rep.

Wittgenstein, Ludwig (1953). *Philosophical Investigations*.

Yang, Yinfei et al. (Apr. 20, 2018). "Learning Semantic Textual Similarity from Conversations". In: *arXiv:1804.07754 [cs]*. arXiv: 1804.07754.

Zhang, Saizheng et al. (Jan. 22, 2018). "Personalizing Dialogue Agents: I have a dog, do you have pets too?" In: *arXiv:1801.07243 [cs]*. arXiv: 1801.07243.

Annex

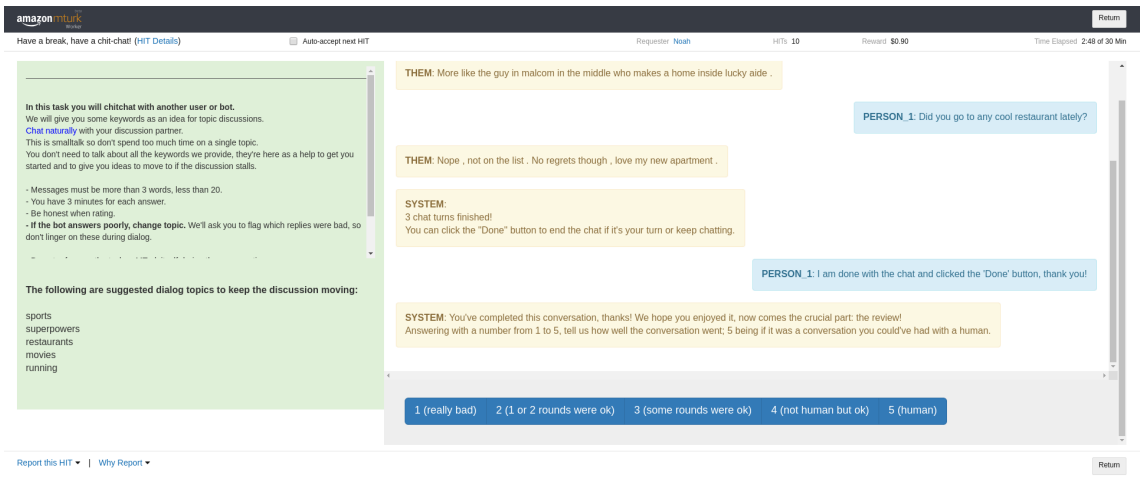

Figure 11: Evaluation in our Mechanical Turk task: rating

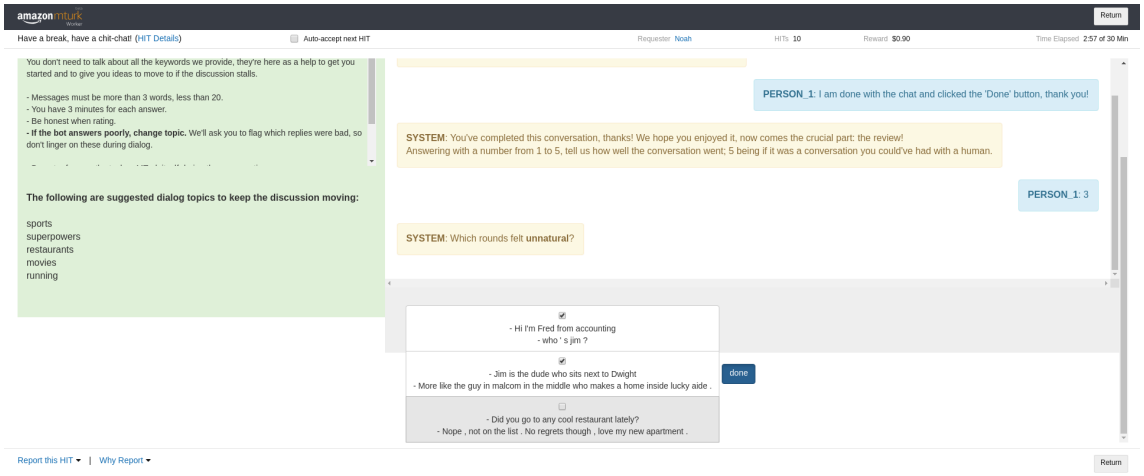

Figure 12: Evaluation in our Mechanical Turk task: selecting bad rounds

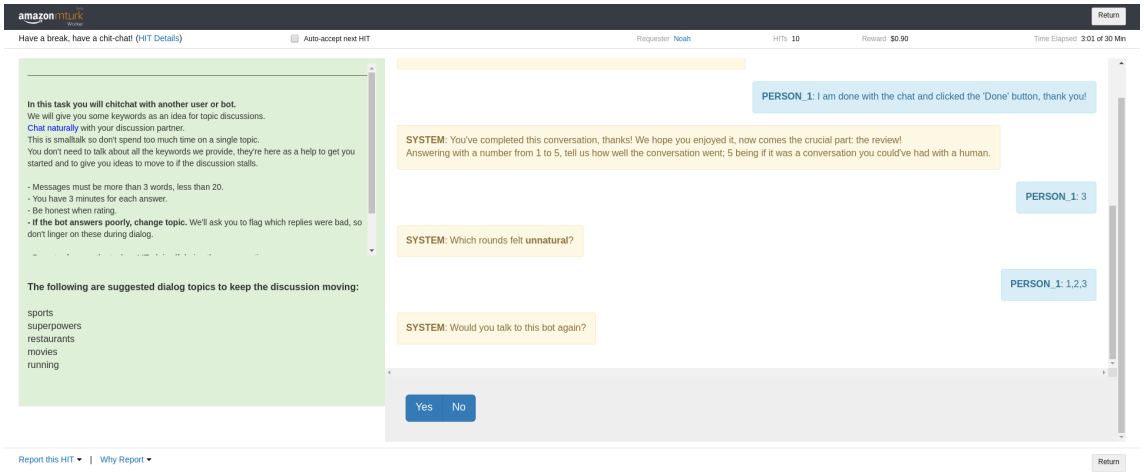

Figure 13: Evaluation in our Mechanical Turk task: a proxy for engagingness

task	metric	pearson	p pvalue	spearman	s pvalue
dialog_permutations-cornell-4	acc	0.534	0.09	0.661	**0.027**
dialog_permutations-pchat-4	acc	0.661	**0.027**	0.642	**0.033**
dialog_permutations-ubuntu-4	acc	0.566	0.07	0.642	**0.033**
ranks-1/100/reddit-random	rank	-0.866	**0.001**	-0.67	**0.024**
ranks-1/1000/reddit-random	rank	-0.866	**0.001**	-0.67	**0.024**
ranks-1/10000/reddit-random	rank	-0.867	**0.001**	-0.67	**0.024**
ranks-1/100000/reddit-random	rank	-0.775	**0.005**	-0.46	0.154
ranks-1/1000000/reddit-random	rank	0.277	0.41	0.287	0.392
ranks-10/100/reddit-random	rank	-0.866	**0.001**	-0.67	**0.024**
ranks-10/1000/reddit-random	rank	-0.866	**0.001**	-0.67	**0.024**
ranks-10/10000/reddit-random	rank	-0.867	**0.001**	-0.67	**0.024**
ranks-10/100000/reddit-random	rank	-0.775	**0.005**	-0.46	0.154
ranks-10/1000000/reddit-random	rank	0.277	0.41	0.287	0.392
ranks-100/1000/reddit-random	rank	-0.866	**0.001**	-0.67	**0.024**
ranks-100/10000/reddit-random	rank	-0.867	**0.001**	-0.67	**0.024**
ranks-100/100000/reddit-random	rank	-0.775	**0.005**	-0.46	0.154
ranks-100/1000000/reddit-random	rank	0.277	0.41	0.287	0.392
ranks-1000/10000/reddit-random	rank	-0.867	**0.001**	-0.67	**0.024**
ranks-1000/100000/reddit-random	rank	-0.775	**0.005**	-0.46	0.154
ranks-1000/1000000/reddit-random	rank	0.277	0.41	0.287	0.392
ranks-1/100/reddit-random	uconf	-0.567	0.069	0.241	0.474
ranks-1/1000/reddit-random	uconf	-0.164	0.63	0.228	0.501
ranks-1/10000/reddit-random	uconf	0.099	0.772	0.255	0.449
ranks-1/100000/reddit-random	uconf	0.152	0.656	0.26	0.441
ranks-1/1000000/reddit-random	uconf	0.175	0.606	0.26	0.441
ranks-10/100/reddit-random	uconf	-0.567	0.069	0.241	0.474
ranks-10/1000/reddit-random	uconf	-0.164	0.63	0.228	0.501
ranks-10/10000/reddit-random	uconf	0.099	0.772	0.255	0.449
ranks-10/100000/reddit-random	uconf	0.152	0.656	0.26	0.441
ranks-10/1000000/reddit-random	uconf	0.175	0.606	0.26	0.441
ranks-100/1000/reddit-random	uconf	-0.164	0.63	0.228	0.501
ranks-100/10000/reddit-random	uconf	0.099	0.772	0.255	0.449
ranks-100/100000/reddit-random	uconf	0.152	0.656	0.26	0.441
ranks-100/1000000/reddit-random	uconf	0.175	0.606	0.26	0.441
ranks-1000/10000/reddit-random	uconf	0.099	0.772	0.255	0.449
ranks-1000/100000/reddit-random	uconf	0.152	0.656	0.26	0.441
ranks-1000/1000000/reddit-random	uconf	0.175	0.606	0.26	0.441
ranks-1/100/reddit-random	nconf	**0.017**	0.961	0.255	0.449
ranks-1/1000/reddit-random	nconf	0.281	0.402	0.519	0.102
ranks-1/10000/reddit-random	nconf	0.268	0.425	0.369	0.264
ranks-1/100000/reddit-random	nconf	0.241	0.476	0.26	0.441
ranks-1/1000000/reddit-random	nconf	0.185	0.586	0.296	0.377
ranks-10/100/reddit-random	nconf	**0.017**	0.961	0.255	0.449
ranks-10/1000/reddit-random	nconf	0.281	0.402	0.519	0.102
ranks-10/10000/reddit-random	nconf	0.268	0.425	0.369	0.264
ranks-10/100000/reddit-random	nconf	0.241	0.476	0.26	0.441
ranks-10/1000000/reddit-random	nconf	0.185	0.586	0.296	0.377
ranks-100/1000/reddit-random	nconf	0.281	0.402	0.519	0.102
ranks-100/10000/reddit-random	nconf	0.268	0.425	0.369	0.264

ranks-100/100000/reddit-random	nconf	0.241	0.476	0.26	0.441
ranks-100/1000000/reddit-random	nconf	0.185	0.586	0.296	0.377
ranks-1000/10000/reddit-random	nconf	0.268	0.425	0.369	0.264
ranks-1000/100000/reddit-random	nconf	0.241	0.476	0.26	0.441
ranks-1000/1000000/reddit-random	nconf	0.185	0.586	0.296	0.377
ranks-1/100/reddit-random	acc	0.731	**0.011**	0.733	0.01
ranks-1/1000/reddit-random	acc	0.646	**0.032**	0.633	**0.036**
ranks-1/10000/reddit-random	acc	0.591	0.056	0.624	0.04
ranks-1/100000/reddit-random	acc	0.575	0.064	0.624	0.04
ranks-1/1000000/reddit-random	acc	0.107	0.755	0.257	0.445
ranks-10/100/reddit-random	acc	0.833	**0.001**	0.724	**0.012**
ranks-10/1000/reddit-random	acc	0.746	**0.008**	0.752	**0.008**
ranks-10/10000/reddit-random	acc	0.652	0.03	0.633	**0.036**
ranks-10/100000/reddit-random	acc	0.543	0.084	0.56	0.073
ranks-10/1000000/reddit-random	acc	0.281	0.402	0.487	0.128
ranks-100/1000/reddit-random	acc	0.835	**0.001**	0.724	**0.012**
ranks-100/10000/reddit-random	acc	0.747	**0.008**	0.752	**0.008**
ranks-100/100000/reddit-random	acc	0.587	0.058	0.688	**0.019**
ranks-100/1000000/reddit-random	acc	0.529	0.094	0.51	0.109
ranks-1000/10000/reddit-random	acc	0.836	**0.001**	0.724	**0.012**
ranks-1000/100000/reddit-random	acc	0.711	**0.014**	0.679	**0.022**
ranks-1000/1000000/reddit-random	acc	0.319	0.34	0.538	0.088

Table 11: Tasks and human evaluation correlations, p-values lower than 0.05 are in bold.

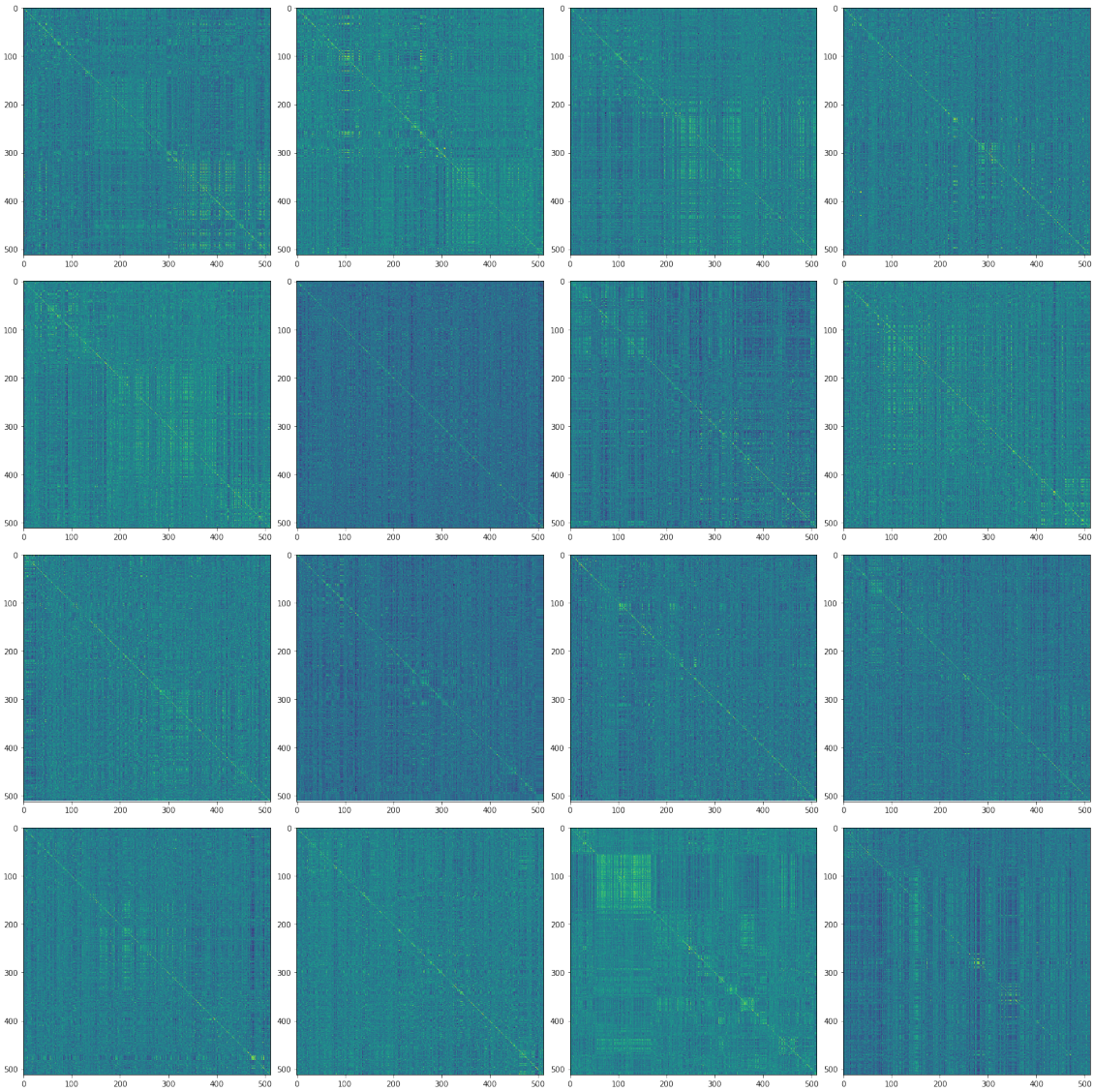

Figure 14: Context-answer similarity in non shuffled batches.

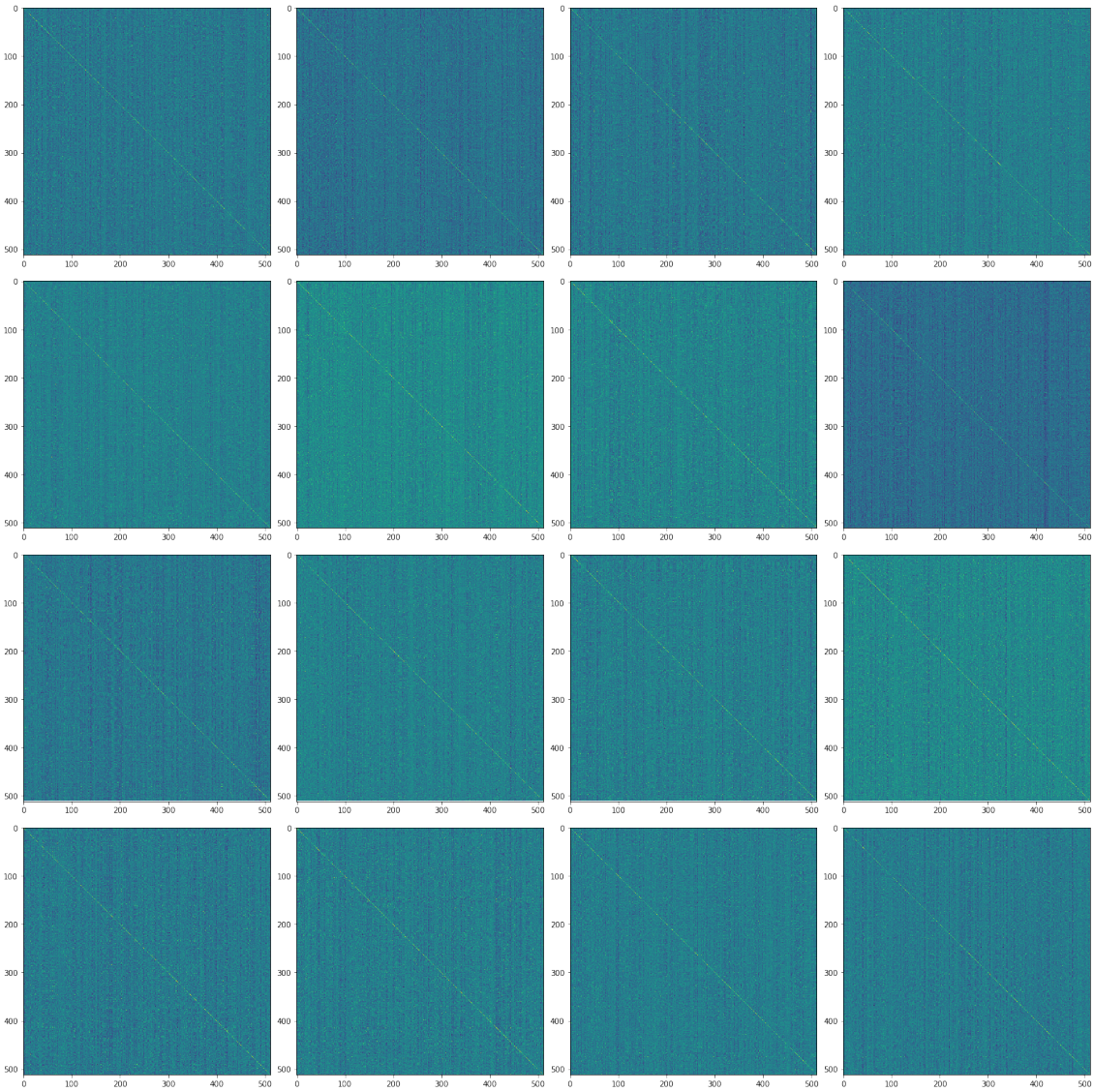

Figure 15: Context-answer similarity in shuffled batches.