E.T.: Entity-Transformers
Coreference augmented Neural Language Model for richer mention representations via Entity-Transformer blocks

Nikolaos Stylianou
Aristotle University of Thessaloniki
School of Informatics
Greece
nstylia@csd.auth.gr

Ioannis Vlahavas
Aristotle University of Thessaloniki
School of Informatics
Greece
vlahavas@csd.auth.gr

Abstract

In the last decade, the field of Neural Language Modelling has witnessed enormous changes, with the development of novel models through the use of Transformer architectures. However, even these models struggle to model long sequences due to memory constraints and increasing computational complexity. Coreference annotations over the training data can provide context far beyond the modelling limitations of such language models. In this paper we present an extension over the Transformer-block architecture used in neural language models, specifically in GPT2, in order to incorporate entity annotations during training. Our model, GPT2E, extends the Transformer layers architecture of GPT2 to Entity-Transformers, an architecture designed to handle coreference information when present. To that end, we achieve richer representations for entity mentions, with insignificant training cost. We show the comparative model performance between GPT2 and GPT2E in terms of Perplexity on the CoNLL 2012 and LAMBADA datasets as well as the key differences in the entity representations and their effects in downstream tasks such as Named Entity Recognition. Furthermore, our approach can be adopted by the majority of Transformer-based language models.

1 Introduction

Language modelling is the task of transforming individual words into vector representations based on the context they appear in. Hence, distant term dependencies are an inherited issue within the task. Language models always seek for smart approaches towards incorporating context from longer distances as it allows for better representations compared to their limited context counterparts. Intuitively, imagine attempting to start reading a novel series from the second book onward, with no information about the first. The amount of information previously missed is something that cannot be acquired. However, this is the case with most language models. While an understanding of the words is present due to the contextual information at each word's occurrence, entity information that are in distant text are lost or not transferred.

Until recently, Recurrent Neural Networks (RNNs), and specifically Long Short-Term Memory (LSTM) networks, have been the core of all the state-of-the-art approaches (McCann et al., 2017; Peters et al., 2018). Thanks to the Transformers architecture (Vaswani et al., 2017), through the use of attention mechanisms, models such as XLNet (Yang et al., 2019), GPT (Radford et al., 2019) and BERT (Devlin et al., 2019) can account for even longer sequences. However, the computational limitations of the multi-head attention in the architecture make it hard to increase the contextual information in such models (Tay et al., 2020). As a result, research has been focused on introducing variations to the transformer architecture, with focus on the multi-head attention mechanism, in order to alleviate part of the computational cost and increase the contextual information available to models.

In this paper we present a novel approach, that makes use of coreference information during training a language model via our Entity-Transformer architecture, which extends the original Transformer block in Transformer-Based language models. To that end, we incorporate the important entity information

This work is licensed under a Creative Commons Attribution 4.0 International License. License details: http://creativecommons.org/licenses/by/4.0/.

Proceedings of the 3rd Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC 2020), pages 1–10,
Barcelona, Spain (online), December 12, 2020.

that would otherwise be unreachable for the model. As a result, we effectively boost the representations of the entity mentions, where entity information is present, without hindering the performance of the language model where entities are not present.

In our experiments, we extend the GPT2 architecture to formulate our model, named GPT2E and train it on the CoNLL-2012 dataset (Pradhan et al., 2012) using the annotated coreference information. We evaluate the model's performance in terms of Perplexity on the ConLL 2012 and the LAMBADA (Paperno et al., 2016) datasets and showcase the effects of such training on the word representations as well as on the downstream task of Named Entity Recognition (NER) using the CoNLL 2012 dataset. To that end, we compare GPT2E's performance to a base model (GPT2) when trained on the same data, to highlight the effects of coreference information when paird with our Entity-Transformer architecture.

2 Related Work

In the last decade, the field of Neural Language Modelling has witnessed enormous changes. With pretrained neural language models being the current go-to approach in all NLP reserach, a variety of methods models have been developed. We distinguish two major categories:

General purpose language models. Steady improvements have been achieved to this field with the use of deep RNNs and pre-training on a large number of training data (McCann et al., 2017; Peters et al., 2018). With Transformers, language models have been able to capture longer linguistic structures without the use of RNNs and surpass their RNN counterparts by a big margin (Radford et al., 2018; Devlin et al., 2019). Recent research has focused on ways of taking advantage of more context (Yang et al., 2019; Fan et al., 2020) and introducing effective methodologies to scale up the models and train them (Radford et al., 2019; Shoeybi et al., 2019; Rosset, 2019; Brown et al., 2020).

Language modelling with entity decisions. YangLM (Yang et al., 2017) was the first to incorporate entity decisions to a language model by introducing learnable entity embeddings. Alternative entity handling mechanisms are introduced in both EntityNLM (Ji et al., 2017) and SetLM (Kunz and Hardmeier, 2019) in addition to a length variable for EntityNLM. All of the aforementioned approaches are RNN-based and hence their performance is expected to be sub-par to Transformer based models. Furthermore, (Kunz and Hardmeier, 2019) concludes that language models handling entity decisions do not improve in performance with the addition of more hidden units and that the source data is of limited number and of specific genre which do not highlight the benefits of explicit entity information. Clark et al. (2019), through attention head probing, experimentally proves that BERT does model anaphoric phenomenon in the form of antecedent selection, with attention heads directly attending to the respective mention's antecedent. However, these information are not explicitly used to further enhance the model. Furthermore, ERNIE (Zhang et al., 2019), which uses knowledge graphs to infuse entity information to the model, only does so for named entities, completely ignoring pronouns and nominal mentions.

3 Our approach

In order to incorporate coreference information to a language model, we require training and testing data with entity information present and a mechanism to handle existing and non-existing entities. To that end, our proposed model, GPT2E, is based on the GPT2 language model, with changes to the Transformer block and an entity handling mechanism, which are described in the following subsections. As a result, GPT2E is a combination of multi-layer Entity-Transformer decoder blocks. The model applies multi-headed self-attention operations over the input tokens, position-wise feed-forward transformations, and entity-based attention operations. The model architecture can be described as follows:

$$\begin{aligned} h_0 &= UW_e + W_p \\ h_l &= \texttt{entity\_transformer\_block}(h_{l-1}, E) \forall_i \in [1, n] \\ P(u) &= \texttt{softmax}(h_n W_e^T) \end{aligned} \tag{1}$$

where $U = (u_{-k}, \ldots, u_{-1})$ is the context vector of tokens, n is the number of layers, W_e is the

token embedding matrix, W_p is the position embedding matrix and E is the context vector of entity representations.

3.1 Entity-Transformer block

Entity-Transformer (ET) blocks are extensions of the transformer blocks used in GPT2, designed to handle entities in the form of vectors of shape $E_i \in \mathbb{R}^{1 \times d_{embd}}$, where d_{embd} is the embedding dimension the model outputs. Effectively, the entity representations are used directly inside the ET blocks.

The input representation first goes through a layer normalization (Ba et al., 2016) and a masked multi-head self attention layer (Vaswani et al., 2017), followed by a residual connection (He et al., 2016). The output of the residual connection is then used in a layer normalization and position-wise feed foward layer followed by another residual connection. The final residual output is used in the entity attention layer before it is forwarded outside of the Entity-Transformer block.

The entity attention layer is an adaptation of the masked multi-head self attention layer which considers Entities (E) as the Key (K) value in the Query (Q), Key (K), Value (V) attention mechanism scheme. The architecture of the Entity-Transformer blocks and the entity attention mechanism used are shown in Figure 1.

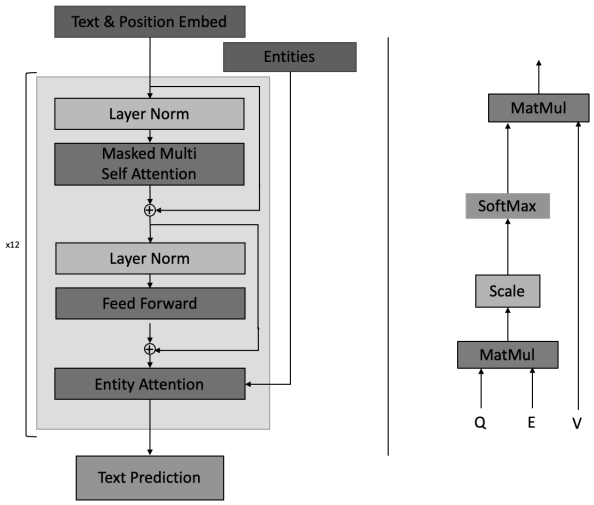


Figure 1: **(left)** Entity-Transformer Block **(right)** Entity Attention mechanism

3.2 Entity handling mechanism

We maintain a persistent set of entities $\mathcal{E}$, that holds the hidden representation of the last entity's mention from our model. Each entity representation E_i is initialised as a vector of ones, which allows for minimal noise in the first occurrence of the entity. Tokens that are not part of the entity mention have a consistent entity representation $E_\emptyset$, as a vector of ones, similar to unseen entity mentions.

During each training step, E_i takes the latest value of the respective entity's latest hidden representation from $\mathcal{E}$ and is updated to the new value at the end of each step. These entity representations are handled with the use of Entity-Transformer blocks. The final hidden representation of the input token, after it is affected by the previous entity representation E_i, is considered to be the new entity representations and replaces E_i in $\mathcal{E}$.

4 Experiments

Our approach is evaluated in two steps. First we evaluate our GPT2E language model, in comparison with a GPT2 model, trained on CoNLL 2012 and evaluated on both CoNLL 2012 and LAMBADA datasets. We then use the trained models to extract word representations for entity mentions based on the coreference annotations in text and measure the differences of such representations. For NER, we use the language models to extract word representatios and train the same baseline model on the CoNLL 2012 dataset.

4.1 Setup

In our experiments we use the GPT2-small configuration with 117M parameters, 12 heads and 12 layers for both GPT2 and GPT2E. Both models use a Byte-Pair Encoder to process the input, a learning rate of 2e-5 and train for 10e5 steps, with validation every 10e3 steps. We use a batch size of 1, to highlight the

effect of entity updates in the system, as the entity representations are only updated at the end of each training step.

After training, we compute the differences between the representations of all entity mentions in the coreference clusters as derived from GPT2 and GPT2E. Consequently, we conduct experiments with no contextual information for each word and we also distinguish the results between using and not using entity information. We perform these experiments separately for all entities in the dataset and present the average score for different type of words based on their part-of-speech tags.

The NER models are based on the Lample et al. (2016) architecture. However, our models use only word embeddings from the pre-trained GPT2 and GPT2E models respectively, removing the character embeddings to eliminate any information input apart from the coreference-trained representations. We use a hidden size of 512 for the Bidirectional LSTMs, 0.5 dropout (Srivastava et al., 2014) between layers and a learning rate of 0.0001 with 0.9 decay per epoch with Adam (Kingma and Ba, 2014). We trained our models for 20 epoches, with early stopping and a batch size of 32.

All the experiments were run on a computer with a single Titan V 12GB graphics card, 32GB of memory and an Intel i7-8700 processor.

4.2 Datasets and Preprocessing

We chose the English CoNLL-2012 dataset for training, which is based on the OntoNotes 5.0 corpus (Weischedel et al., 2011) and contains over 1.3 million words with 35,143 entity mentions in the training set and 170 thousand words with 4,532 entity mentions in the test set making it the most suitable dataset for training a language model with coreference annotations. In the dataset common nouns, pronouns and proper nouns contribute 90% of the words in both train and test English sets. For our out of domain evaluation we chose the LAMBADA dataset. This choice was based on the premise that the dataset is primarly used for word predictions requiring broad discourse context and that the target words are mostly proper nouns and common nouns (85% fo the total target words). As a result, we expect that the importance of an entity-centric language model would be better displayed in such a scenario.

As we utilize the CoNLL-2012 dataset for both the Language Modelling task and the NER task, we formulate the data in two different ways.

Table 1: Data example from the CoNLL 2012 dataset, as formated for the task.

$X_{1:11}$	“	The	U.S.	underestimated		Noriega	all	along	”	says	Ambler	Moss
$E_{1:11}$	∅	73	73	∅		82	∅	∅	∅	∅	50	50
$X_{12:23}$	a	former	Ambassador	to	Panama	.	“	He	has	mastered	the	art
$E_{12:23}$	50	50	50	50	50	∅	∅	82	∅	∅	∅	∅

For Language Modelling, we formulate our data in a similar manner with Ji et al. (2017), as seen in Table 1. Specifically, for each token we also introduce a second variable “E” which indicates the entity in which the token is part of, using the gold coreference annotations, with a special “∅” for tokens that are not part of an entity. For the CoNLL dataset, we populate E with the golden entities from the coreference resolution shared task. For the LAMBADA dataset we use the ∅ for all tokens. In comparison to the original data formulation described in Ji et al. (2017), we opted to not use the L variable to denote the entity length (i.e. the number of remaining tokens in the entity mention) as it's main use is enable entity mention prediction, which we do not attempt at this stage. We use Byte Pair Encoding (BPE) (Sennrich et al., 2016) for the final input representation of the word instances, similar to GPT2.

For NER, we formulate the data in a IOB format to facilitate a similar model architecture as described in Lample et al. (2016), using the gold named entities of the dataset, including nested entities.

5 Results

To evaluate the results of our Entity-Transformers architecture and the effects of corereference annotations to language modelling, we measure the change in performance of the language model using

Perplexity (PPL). Furthermore, we compute the average difference of the representations between mentions of the same entity of the GPT2E model, between each entity mention between GPT2 and GPT2E and between non-entity mentions of the same words using cosine similarity. Furthermore, we use micro-average Precision, Recall and F1 scores for the evaluation of our NER models.

For Language modelling, Table 2, shows the training and validation losses of GPT2 and GPT2E, as well as the Perplexity of the models after 10e5 training steps. The gradual changes in training and validation losses, measured every 10e3 steps, are illustrated in Figures 2 & 3 with GPT2 model in orange and GPT2E model in blue colours respectively. Similarly, Table 3 highlights the performance difference between the two trained models on the LAMBADA dataset. As both models are trained on a very limited dataset compared to other language models, we are not comparing performance in terms of accuracy.

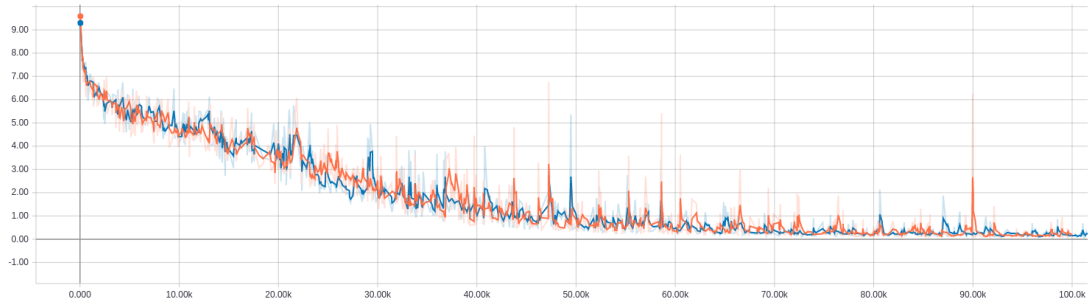


Figure 2: Training loss per step on the CoNLL 2012 dataset.

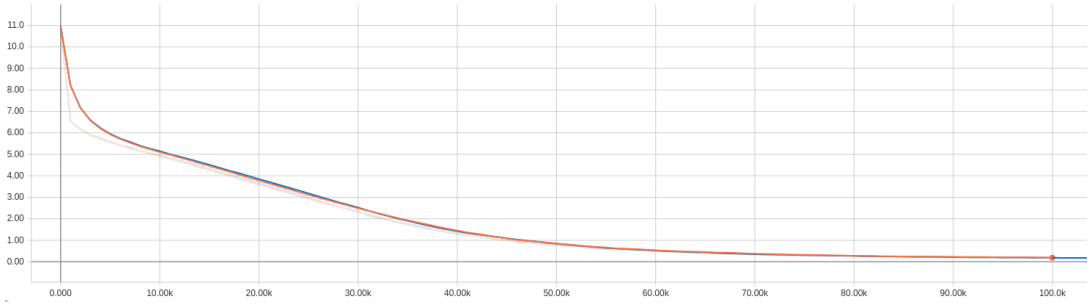


Figure 3: Validation loss per step on the CoNLL 2012 dataset.

Table 2: Perplexity and Validation loss on the CoNLL 2012 dataset

Process	GPT2E			GPT2		
	PPL	Loss	Time per step	PPL	Loss	Time per step
Training	5.52	1.71	0.290s	4.80	1.57	0.298s
Validation	1.20	0.187	0.290s	1.19	0.184	0.298s

Table 3: Perplexity performance on the LAMBADA dataset

Model	Perplexity
GPT2E	196.81
GPT2	219.97

In terms of Perplexity, the models show similar performances on the CoNLL 2012 dataset, while having a slight advantage at the LAMBADA dataset. The slight improvement in Perplexity of the GPT2E model over the GPT2 on the LAMBADA dataset is attributed to the target words' part-of-speech type. As described in Section 4.2, the target words of the LAMBADA dataset are mostly proper nouns and common nouns and the majority of the training mentions in the CoNLL-2012 dataset are of the same type. This behaviour is consistent with the expectations of the performance of an entity-centric language model. Both GPT2 and GPT2E models show a remarkably low Perplexity compared to EntityNLM, YangLM and SetLM of reported Perplexity 161.64, 114 and 107 respectively. However, these language models are RNN based, and gap between them is attributed to the Transformers architecture and the

relatively small size of the CoNLL-2012 dataset. The added complexity of calculating the entity representations and using the Entity-Transformer blocks is contributing to 0.008 seconds per step in both training and evaluation, adding up to an additional 12 min and 6 seconds, a 2% increase in time for the complete training process.

To compare the changes in the entity mention representations when using coreference information during training we conducted a series of experiments, taking into account the existence or absence of coreference annotation. Specifically, for both models, for each entity we calculate the average similarity of its mentions with the other entity mentions, with and without the use of entity representations for GPT2E, and the average similarity between the entity representation and the entity mentions. We have limited the scope of the comparisons, using part-of-speech tags, to only nouns and proper nouns, as these will be the words that will be affected the most by our changes, given the dataset statistics presented in Section 4.2. Similarly, we calculated the average cosine similarity between the pronoun's representations of the two models as well as the differences between the two when entity representations are present.

Table 4: Cosine similarity of mention representations and their entities in different scenarios

Experiments	GPT2E without Entities	GPT2E with Entities	GPT2
Average mention similarity NN,NNS,NNP,NNPS	0.7117	0.7117	0.6971
Average entity similarity NN,NNS,NNP,NNPS	0.0489	0.0513	-0.0164
Average mention similarity PRP,PRP$	0.8250	0.8250	0.7928
Average entity similarity PRP,PRP$	0.0619	0.0566	-0.0173

Based on the results displayed in Table 4, we can infer that the mentions maintain their similarity when the coreference information are used during inference, while also have a higher average similarity than the respective mentions of the model trained without coreference annotations. However, taking into account the changing similarity scores between the entity representations and the entity mentions when we use coreference information during inference, we can conclude that there is a constant change to the representations. In the case of nouns and pronouns, that change brings the representations closer while in pronouns it has the opposite effect. Individual visual representations of the embeddings for GPT2E and GPT2 and a comparative visual representation between the two are included in the appendix section.

Table 5: NER performance using GPT2 and GPT2E representations as input.

Labels	GPT2			GPT2E		
	F1	Prec	Recall	F1	Precision	Recall
PERSON	48%	95.5%	32.5 %	51.5%	94%	35.5%
PRODUCT	8%	33%	4.5 %	23.5%	90%	13.5%
EVENT	23%	83.5%	13.5%	15%	75%	8.5%
CARDINAL	28%	81.5%	17.5%	34%	75%	23%
NORP	44.5%	72.5%	36%	48%	79%	39.5%
Overall	54%	87%	39%	57%	88%	42%

The NER model, trained using word representations from GPT2E, achieved a mean average 3% F1 increase than the one trained with GPT2 word representations. We highlight four named entities in Table 5, which showed the biggest differences between the two trained models. Specifically, we observe that the named entities of PERSON and PRODUCT, which would be directly affected by the anaphoric information in the training process, showed the greatest increase and contributed the most to the per-

formance boost. Subsequently, EVENT entities were more commonly mislabelled while using GPT2E representations. This behaviour is credited to the use of LOCATION terms to describe events (e.g. "the Guangzhou Fair") and to generic event terms that refer to different entities based on their context (e.g. "new year" can refer to a different year) which the baseline model was unable to handle correctly when the word representations were affected by entity information.

6 Discussion and Conclusions

In this paper we demonstrated a novel architecture to use coreference information in transformer-based neural language models in order to create richer representations and its effects on downstream tasks. We introduced an extension over the Transformer blocks of GPT2, labeled Entity-Transformer, that integrates coreference information to each entity mention. To that end, we also created an entity handling mechanism to create and update entity representations. Furthermore, as our proposed architecture extends over the basic Transformer block, it can be easily adapted to other Transformer-based language models, such as BERT, and also enables further research for Transformer-based language models with explicit entity decisions which have far outperformed their RNN counterparts.

In our experiments we showcased that in terms of Language modelling, both GPT2E and GPT2, when trained on the same data, have indistinguishable performance in terms of Perplexity and GPT2E has a small computational cost that translates into a slightly longer training time. However, the difference in the similarity between entity mention representations suggests that fewer iterations and mentions of each word are required to achieve the results, assuming a large enough number of mentions. This is due to the extended contextual information present at each mention occurrence, in the form of entity representations, used when training the model. What is more, the differences in these representations directly translates to an increase in tasks such as Named Entity Recognition. As coreference is ever-present in natural language, with a better ability for a language model to understand and utilize the anaphoric phenomenon in text, we expect an increased performance in other tasks such as summarization and natural language inference.

In order for language models to use coreference information, there are two requirements that need to be met. First, the models need to replace the Transformer blocks with the Entity-Transformer blocks introduced and also adopt the entity handling mechanism to make use of entity information. Second, annotated coreference information are required throughout the training corpus. While the changes described for the language models are trivial, language models require an enormous amount of training data, making it impossible to manually annotate coreference information. However, the entity handling mechanism we introduced is not affected by the lack of entity information in the training and is only boosted by the existence of them. As a result, even sparse annotations of high confidence will allow for improvements in the representations.

In the future, we plan to extend our work, using noisy annotation provided by pretrained coreference resolvers so that we can train GPT2E to the WikiText dataset (Merity et al., 2018), creating a comparable model with the original GPT2 and other state-of-the-art language models in a wider range of tasks. Furthermore, we aim to expand the abilities of our current approach to be able to make explicit entity decisions, similar to the previously cited work. For that purpose, attention head probing techniques, which have been found to model some anaphoric phenomena (Clark et al., 2019), and transfer learning through weight initialization from a pre-trained GPT2 model will be investigated as they can contribute to significant improvements while needing less annotated training data.

Acknowledgements

This research is co-financed by Greece and the European Union (European Social Fund- ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning" in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research" (MIS-5000432), implemented by the State Scholarships Foundation (IKY).

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy, August. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Angela Fan, Thibaut Lavril, Edouard Grave, Armand Joulin, and Sainbayar Sukhbaatar. 2020. Accessing higher-level representations in sequential transformers with feedback memory. *arXiv preprint arXiv:2002.09402*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A. Smith. 2017. Dynamic entity representations in neural language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1830–1839, Copenhagen, Denmark, September. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jenny Kunz and Christian Hardmeier. 2019. Entity decisions in neural language modelling: Approaches and problems. In *Proceedings of the Second Workshop on Computational Models of Reference, Anaphora and Coreference*, pages 15–19.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California, June. Association for Computational Linguistics.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. An analysis of neural language modeling at multiple scales. *arXiv preprint arXiv:1803.08240*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Ngoc Quan Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1525–1534, Berlin, Germany, August. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

C Rosset. 2019. Turing-nlg: A 17-billion-parameter language model by microsoft. *Microsoft Blog*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using gpu model parallelism. *arXiv preprint arXiv:1909.08053*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. 2020. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

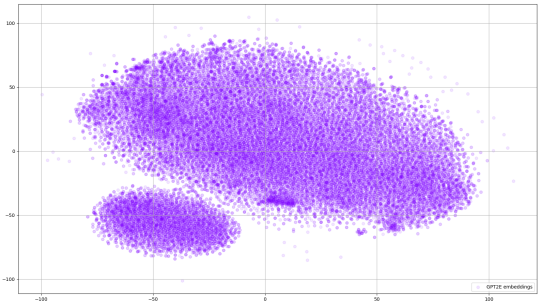
Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, and Nianwen Xue. 2011. Ontonotes: A large training corpus for enhanced processing. *Handbook of Natural Language Processing and Machine Translation. Springer*, page 59.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1850–1859, Copenhagen, Denmark, September. Association for Computational Linguistics.

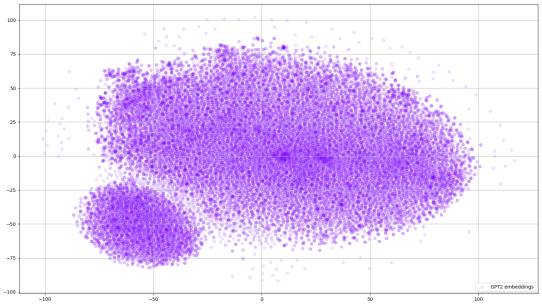
Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy, July. Association for Computational Linguistics.

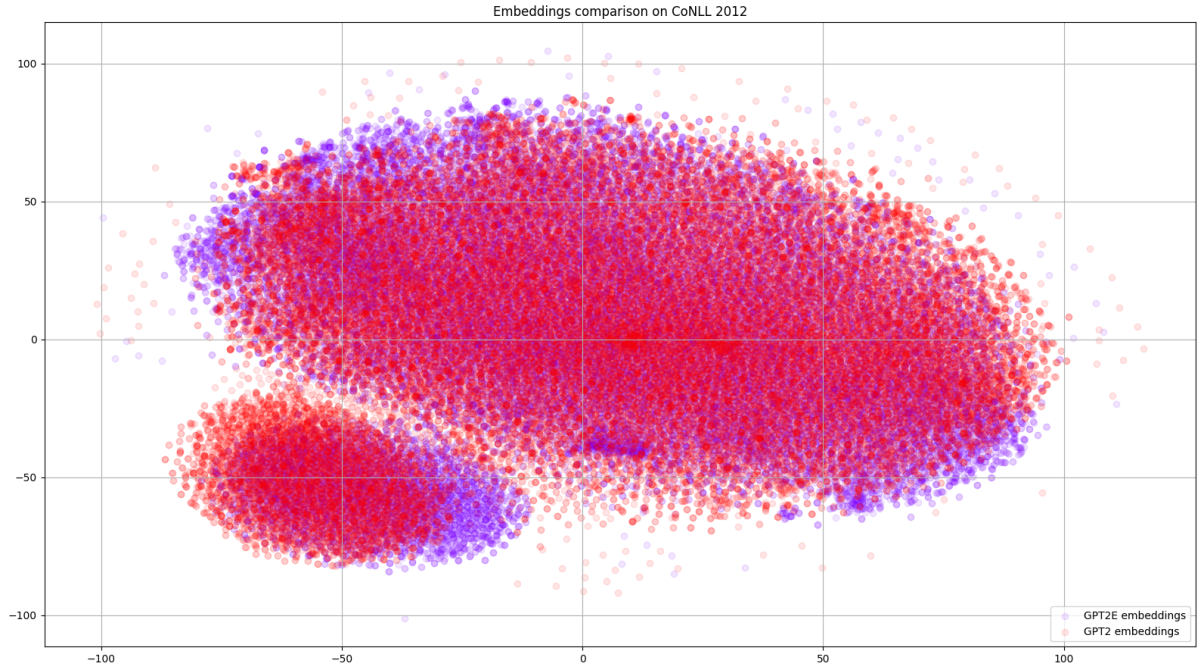
Appendix A. Embeddings visualizations



(a) GPT2E embeddings.



(b) GPT2 embeddings.



(c) Embeddings comparison between GPT2E and GPT2.

Figure 4: Visualization of the word representations of (a) GPT2E and (b) GPT2E and (c) comparison between the two, trained on the CoNLL2012 dataset, using t-SNE.