A Comprehensive Comparison of Word Embeddings in Event & Entity Coreference Resolution.

Judicael Poumay
ULiege/HEC Liege
Rue Louvrex 14, 4000 Liege, Belgium
judicael.poumay@uliege.be

Ashwin Ittoo
ULiege/HEC Liege
Rue louvrex 14, 4000 Liege, Belgium
ashwin.ittoo@uliege.be

Abstract

Coreference Resolution is an important NLP task and most state-of-the-art methods rely on word embeddings for word representation. However, one issue that has been largely overlooked in literature is that of comparing the performance of different embeddings across and within families in this task. Therefore, we frame our study in the context of Event and Entity Coreference Resolution (EvCR & EnCR), and address two questions : 1) Is there a trade-off between performance (predictive & run-time) and embedding size? 2) How do the embeddings' performance compare within and across families? Our experiments reveal several interesting findings. First, we observe diminishing returns in performance with respect to embedding size. E.g. a model using solely a character embedding achieves 86% of the performance of the largest model (Elmo, GloVe, Character) while being 1.2% of its size. Second, the larger model using multiple embeddings learns faster overall despite being slower per epoch. However, it is still slower at test time. Finally, Elmo performs best on both EvCR and EnCR, while GloVe and FastText perform best in EvCR and EnCR respectively.

1 Introduction

Coreference Resolution (CR) is an important NLP task. It can be subdivided into Event and Entity Coreference Resolution (EvCR and EnCR). These tasks serves as the basis for several downstream applications such as information extraction, text summarization, machine translation and text mining (Humphreys et al., 1997; Azzam et al., 1999; Miculicich Werlen and Popescu-Belis, 2017; Su et al., 2008).

State-of-the-art methods for CR(Barhom et al., 2019; Lee et al., 2017; Joshi et al., 2019) rely on various word embeddings for word representation. These embeddings are organized into three families: *static*, *contextual* and *character* embeddings (Almeida and Xexéo, 2019; Liu et al., 2020; dos Santos and Zadrozny, 2014), each differing in size. Contextual embeddings are larger (1024) compared to the other families (usually 300 for static and 50 for character). They also tend to outperform the other families in most tasks but lead to larger and heavier models (Devlin et al., 2019; Peters et al., 2018). We are thus confronted with a trade-off of performance (predictive & run-time) vs. dimensionality. Moreover, embeddings also differ within families which also leads to differences in predictive performance.

Several studies investigated how different embeddings influence the predictive performance in different tasks (Berardi et al., 2015; Gromann and Declerck, 2018; Joshi et al., 2019; Li et al., 2018). However, the two aforementioned issues of the performance vs. dimensionality trade-off and performance variations within and across embedding families have been overlooked to a large extent, especially in coreference resolution. Literature is still unclear about which embeddings perform best in which tasks, and whether larger, more expressive embeddings should also be preferred or whether some predictive performance can be compromised for improved run time.

Thus, we seek to address two questions in the context of CR: 1) Is there a trade-off between performance (predictive & run-time) and embedding size? 2) How do the embeddings' performance compare within and across families? The current state-of-the-art in EvCR (Barhom et al., 2019) rely on three families of embeddings for word representation, and thus provides a suitable frameworks for addressing our research questions. Starting from the original model of Barhom et al. (2019), we performed various experiments and ablative studies across and within each family of embeddings, resulting in 16 different models. [1]. We compared

[1]The relatively large number of models and experiments is one reason why we preferred to focus on a single task

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2755–2764
November 7–11, 2021. ©2021 Association for Computational Linguistics

their predictive performance, size (number of parameters) , run-time and memory usage.

We discovered high level of diminishing returns in term of predictive performance per embedding. The smallest model (using solely a character embedding (dos Santos and Zadrozny, 2014)) achieves 86% of the performance of the largest model (GloVe (Pennington et al., 2014) , ELMo (Peters et al., 2018), Character embedding) with 1.2% of its size. Hence, incorporating additional embeddings leads to diminishing returns in terms of predictive performance. In addition, we found that size and run-time are weakly correlated: larger (more complex) models can converge faster (number of epochs and total training time) than smaller ones . In terms of predictive performance, we found GloVe and FastText perform best in EvCR and EnCR respectively in their family with ELMo being the best overall. Moreover, we found that the smallest aforementioned model outperforms Word2Vec (∼+10 F1), yielding predictive performance close to the previous state-of-the-art (Kenyon-Dean et al., 2018) in EvCR (68.43 vs 69 F1). Our results can have important implications for practitioners in implementing CR and other NLP models in real-life applications.

2 Background and Related work

2.1 Word embeddings families

Literature generally distinguishes between three families: *static*, *contextual* and *character* embeddings (Almeida and Xexéo, 2019; Liu et al., 2020; dos Santos and Zadrozny, 2014).

Static embeddings, such as word2vec, FastText, and GloVe, create a one-to-one mapping between words and their vector representations. Word2vec (Mikolov et al., 2013) learns through a language modelling task by either learning to predict a word given its context (CBOW) or predict the context given a word (Skip-gram). FastText (Bojanowski et al., 2017) learns sub-words embeddings which are then combined for each word. Finally, GloVe (Pennington et al., 2014) relies on word co-occurrence information. Both Glove and FastText are trained on a Skip-gram task.

Contextual embeddings take into account the context of a given word, i.e. their vector representations changes depending on surrounding words. ELMo is a Bi-LSTM trained on a language modelling task. GPT-2 is similar except that it is unidirectional. Finally, BERT is based on a transformer architecture and trained on a masked language modelling task.

Lastly, *character embeddings* learn vectors based on character sequences (dos Santos and Zadrozny, 2014).

Since their development, word embeddings have been very largely studied (Tan et al., 2015; Chen et al., 2018; Wang et al., 2018; Clark et al., 2019; Tenney et al., 2019) and a complete literature review is out of the scope of our work. Hence, we will focus on studies closest to ours. First, we will review studies on embeddings' performance regardless of the task. Then, we move to our task of interest which is coreference resolution.

2.2 Studies on Embeddings' Performance

Gromann and Declerck (2018) found that FastText (0.812 F1) outperformed Polyglot (0.675 F1) and Word2Vec (0.750 F1) for ontology alignment. They used two ontologies: Global Industry Classification Standard and Industry Classification Benchmark. They also demonstrated the ability of FastText to better handle out-of-vocabulary words.

Berardi et al. (2015) found that Word2Vec (Accuracy (ACC) 43.63%) outperformed polyglot (ACC 4%) and GloVe (ACC 30.21%) on a word analogy test using Wikipedia and a collection of Italian books (mostly novels) as datasets.

Joshi et al. (2019) found that BERT significantly outperformed ELMo on EnCR (+11.5 F1) on the GAP and OntoNotes datasets.

Li et al. (2018) found that GloVe outperformed FastText and Word2Vec on a tweet classification task, especially when trained on specific corpora, viz.CrisisLexT6, CrisisLexT26, and 2CTweets.

2.3 Word embeddings in Coreference Resolution.

Event Coreference Resolution and Entity Coreference Resolution (EvCR and EnCR respectively) are concerned with clustering Event and Entity mentions that refer to the same reality (Barhom et al., 2019; Lee et al., 2017). Figure 1 depicts two event mentions with the same meaning.

Figure 1: Two coreferent event mentions with colors indicating associated coreferent entity mentions.

Events mentions refer to textual representations

of real-life events. As can be seen from Figure 1, events generally consist of a trigger word (most often a verb), such as "launched", and a set of arguments, such as "SpaceX" and "a South Korean Military satellite". Four argument types are generally distinguished: Arg0, Arg1, location, and time, as defined in Barhom et al. (2019), where Arg0 (resp. Arg1) is the closest entity on the left (resp. right) of the trigger word. These arguments are optional and often referred to as entities. The goal of EvCR (and EnCR) is to identify which events (and entities) are coreferent with each other and to cluster them.

We now briefly review studies using word embeddings for EnCR and EvCR.

EnCR : Lee et al. (2017) used GloVe as word representation allied with a Bi-LSTM and attention mechanisms. Their model achieved state-of-the-art (68.8 F1) on the the CoNLL-2012 corpus. As already mentioned, Joshi et al. (2019) reported higher EnCR performance when using BERT compared to ELMo: +3.9 F1 in OntoNotes and +11.5 F1 in GAP.

EvCR : Choubey and Huang (2017) relied on GloVe for EvCR using the ECB+ corpus (Cybulska and Vossen, 2014). They used a joint modelling approach to perform within and cross document EvCR and achieved state-of-the-art performance. The same corpus was employed by Barhom et al. (2019), who proposed an EvCR/EnCR model based on ELMo (Peters et al., 2018), GloVe (Pennington et al., 2014) as well as a fine-tuned character embedding. Similarly, it jointly performs EnCR and EvCR. Their model yielded performance of 79.5 F1 in EvCR.

3 Methodology

3.1 Original model

Our approach is based on the state-of-the-art model of Barhom et al. (2019), which we refer to as the ORIGINAL [2] model. This model consists of two neural networks, which jointly resolve entities and events coreferences. Figure 2 shows the input of both networks. The two event (resp. entity) mentions embeddings are in blue and the green box represents an element-wise multiplication of the mentions. Finally, binary features indicate whether the two encoded mentions have coreferent arguments. The constituents of each mention, i.e. trigger, Arg0, Arg1, Location and time, are represented by a static (GloVe) and a character embedding. The trigger is also represented by a contextual embedding (ELMo). Furthermore, the character embedding is fine tuned during training while the contextual and static embeddings are not.

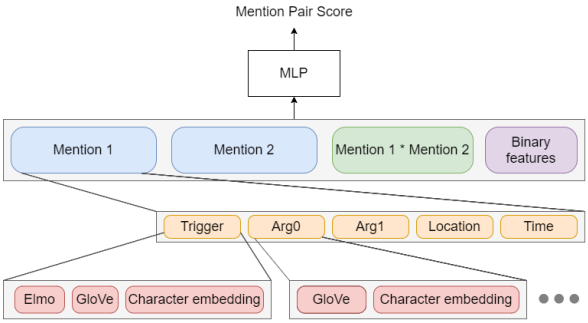

Figure 2: Original input structure of Barhom et al. (2019)'s model.

The input dimensionality is 3*(1024+5*(300+50)) + 200 = 8522, where 1024, 300 and 50 are the dimensions of ELMo, GloVe and the character embeddings, and 200 corresponds to the size of the binary features. This input is then fed into two subsequent ReLU layers with dimensions equal to half the input dimension (4261 neurons each). Since the number of parameters is proportional to the square of the input dimension, we have a model size exceeding 54 million parameters, computed as $(\frac{input^2}{2} + (\frac{input}{2})^2 + \frac{input}{2})$.

3.2 Derived models

The gist of our methodology involves substituting and/or removing specific embeddings from Barhom et al. (2019)'s original model (which uses 3 embeddings : static=GloVe, contextual=ELMo and character), resulting in 16 different models shown in Table 1. In the first group of models, one, two, or three (of the three) embeddings are removed from the original model. In the second group, the static embedding is changed to Word2Vec (Skip-gram) or FastText (other embeddings are either left unchanged or removed). Similarly, in the third group the contextual embedding is changed to BERT or GPT-2 (other embeddings are either left unchanged or removed). Note: in Table 1, gray rows denote identical models.

We implemented our models using Pytorch. Models were trained and tested following Barhom et al. (2019)'s procedure. Pre-trained vectors and models were used for the embeddings. Our code is

[2] MODELNAME denotes a model

available online [3].

Model	Stat.	Ctx.	Char.
Group 1: Across family study			
Original (2019)	GloVe	ELMo	✓
Contextual/Static	GloVe	ELMo	X
Contextual/Char	X	ELMo	✓
Static/Char	GloVe	X	✓
Static	GloVe	X	X
Contextual	X	ELMo	X
Char	X	X	✓
No word embed	X	X	X
Group 2: Within family study: Static			
GloVe	GloVe	ELMo	✓
Word2Vec	Word2Vec	ELMo	✓
FastText	FastText	ELMo	✓
Only GloVe	GloVe	X	X
Only FastText	Word2Vec	X	X
Only Word2Vec	FastText	X	X
Group 3: Within family study: Contextual			
ELMo	GloVe	ELMo	✓
BERT	GloVe	BERT	✓
GPT-2	GloVe	GPT-2	✓
Only ELMo	X	ELMo	X
Only BERT	X	BERT	X
Only GPT-2	X	GPT-2	X

Table 1: List of trained and tested model and their components. Ctx. = Contextual; Stat. = Static; Char. = Character; X/✓indicate absence/presence of an input.

4 Experimentation setup

4.1 Dataset

The dataset we use for our study is ECB+ (Cybulska and Vossen, 2014). Together with EECB (Lee et al., 2012), it is one of the largest datasets for within and cross document EvCR and EnCR (Lee et al., 2012; Barhom et al., 2019). Both EECB and ECB+ are extensions of ECB (Bejan and Harabagiu, 2010) and consist of English Google News documents clustered into topics and annotated for coreference. For more details on the ECB+ corpus statistics, please refer to Barhom et al. (2019).

Other dataset for coreference resolution exist : GAP, OntoNotes, CoNLL 2012, ACE, TAC KBP and MUC. However, the definition of coreference resolution in these corpora do not suits our study and model. For example, GAP is a corpus of ambiguous pronoun-name pairs while ECB+ defines mentions cluster for events and their entities (Joshi et al., 2019). OntoNotes annotates coreferences but does not indicate which mentions is an event and which is an entity. MUC, ACE, and TAC KBP do not provide cross document coreferences(Lu and Ng, 2018). Finally, while CoNLL 2012 defines an event coreference task, events represent only a small portion of the all the coreferent mentions and again it does not provide cross document coreferences (Pradhan et al., 2012). In-depth reviews of the listed datasets are provided in (Stylianou and Vlahavas, 2021; Lu and Ng, 2018; Sukthanker et al., 2018).

4.2 Experiments

We performed three sets of experiments. The first set concerns models of Group 1 (see Table 1). We investigated the impact of removing one, two, or three (of the three) embeddings from the original model. Our aim was to determine the contribution of the different embeddings (static, contextual and character) on the predictive performance of the ORIGINAL model. Thus, the models will have varying sizes, translating into varying run-time and memory requirements. Therefore, for this set of experiments, we also report on model size (number of parameters), run-time (seconds) and memory usage (RAM).

The second (third) set concerns models of Group 2 (Group 3) (see Table 1) and aim at investigating the contributions of static (contextual) embeddings.

For the latter two experiments, we do not consider model size as all possible sizes would have been investigated in group 1. For all experiments, we will report the predictive performance achieved by the various models with the CoNLL F1 and MUC F1 metrics (Moosavi and Strube, 2016).

Following Barhom et al. (2019)'s original paper, we can claim that a difference of 1 point between any two models is significant with a p-value $<$ 0.001. This confirms that our results are statistically sound and not due to randomness.

5 Results

5.1 Results 1: All Embedding Families

As mentioned earlier, our aim was to investigate the contributions of the static (Glove) , contextual (ELMo) and character embedding to the original model's performance via an ablative study. The

[3] github.com/JudicaelPoumay/event_entity_coref_ecb_plus

predictive performance scores (CoNLL/MUC F1) of Group 1 models are in Figure 3, respectively from left to right.

A first observation is that the baseline performance differs between the two measures (CoNLL & MUC F1). This is due to the mention identification effect (Moosavi and Strube, 2016) which makes CoNLL F1 more optimistic than it should be for low performing models. Interestingly, CoNLL seems more pessimistic than MUC for high performing models. Moreover, Barhom et al. (2019)'s model is helped by using gold cluster for within-document entity coreference. This explains the non-zero MUC F1 performance of the baseline on the entity coreference resolution task.

Another important observation is that, when using only two embedding, the STATIC/CHAR model is the one experiencing the largest drop in performance (CoNLL & Event MUC). At the same time, when using only one embedding, the CONTEXTUAL model performs best. It even outperforms the aforementioned model with *two* embeddings: STATIC/CHAR. These results lead us to conclude that the contextual embeddings is the most expressive for this task. This is not surprising since contextual embeddings take context into account while static and character do not.

More interestingly, we note that removing either the static or contextual embedding results in an average performance drop of ∼2.5 and ∼4 CoNLL points respectively (see model CONTEXTUAL/CHAR and STATIC/CHAR). However, when both are removed simultaneously, the performance drops by ∼10 CoNLL points (see model CHAR). That is, the sum of the losses incurred by removing either one of these embeddings (∼6.5) is smaller than the loss (∼10) incurred when both are simultaneously removed. Similarly, adding *any one* embedding to the baseline NO WORD EMBEDDING model significantly improves the latter's performance, in the range of ∼[+27,5 to +34,7].However, if *any one* embedding is removed from the ORIGINAL model, then the latter's performance drops by a much smaller amount,∼[-1,1 to -4]. That is, removing an embedding from the ORIGINAL model does not impact performance in a comparable way as adding an embedding to the baseline model. But performance does drop significantly when all embeddings are removed. In other words, we face diminishing returns in terms of performance per embeddings.

Impact of Dimensionality on Model Size

As mentioned earlier, the model size is related to the square of the input, resulting in more than 54 million parameters in the ORIGINAL model. Thus, an important question is that of whether the gains in performance of such large models outweigh the corresponding increase in size. Our observations in this respect are in Figure 4, depicting the model's respective size and predictive performance. We observed similar diminishing returns when considering performance relative to size, i.e. increasing the model size by incorporating larger, more complex embeddings results in modest performance gains.

The CONTEXTUAL and CHAR models are particularly interesting. The former achieves 96% of the performance of the ORIGINAL model with 14.7% of its size. While the latter, i.e. CHAR, achieves 86% of the performance of the ORIGINAL model's performance, with only 1.2% of its size. Its performance (68.43 F1) is even comparable to that of the previous event coreference resolution state-of-the-art in EvCR (69 F1) (Kenyon-Dean et al., 2018).

Model Size & Run-Time

Our investigations on the influence of model size on run-time and memory usage revealed paradoxical results.[4] They are presented in Figure 5. For the run-time and memory analysis, we focus only on the largest and smallest models to have a better idea of the magnitude of differences and to avoid overcrowding the Figures.

As can be seen, the huge difference in model size (54 Million vs. 0.67 Million), does not translate into equally large the differences in run-time (training & testing) - the run-time reductions afforded by the CHAR model are relatively modest. While the actual reasons deserve further investigation, we can posit that this could be attributed to hardware and software optimization, enabling a high level of parallelization such that larger models run comparably to smaller ones.

Paradoxically, however, the larger ORIGINAL model trains in fewer epochs than the smaller CHAR model (14 vs. 24 respectively). In consequence, it is 21% faster to train overall (68924.8 sec. vs 87587.28 sec. or about 19h9 vs 24h19). These results confirm the observation of Li et al. (2020)

[4]Ran on a Ryzen 5 3600X CPU and a RTX 2070 Super GPU along with 32GB of RAM

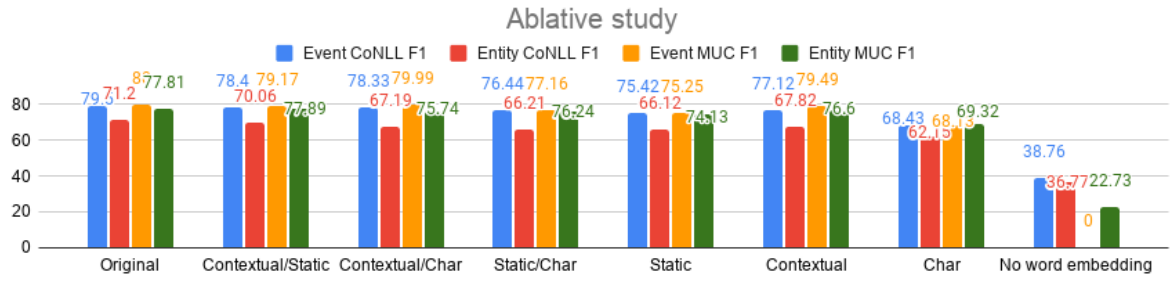

Figure 3: Comparing the predictive performance of the original model (using 3 embeddings) with models where we removed one, two or all three embeddings.

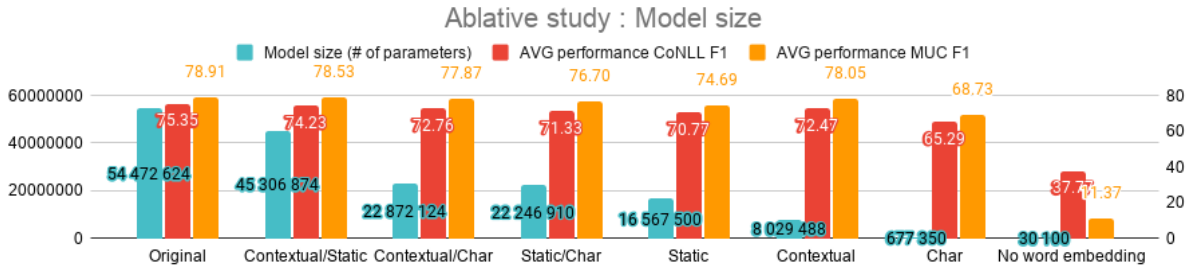

Figure 4: Comparing the size and predictive performance of the original model (using 3 embeddings) with models where we removed one, two or all three embeddings. The size of each model is the number of neural connections.

that larger models tend to converge faster. One possible explanation could be that larger models have to optimize a error surface of higher dimensionality, leading to more possible paths for gradient descent, some of which might lead to convergence more rapidly. Thus, although adding more embedding in the model results in diminishing returns in term of predictive performance, it can lead to faster training. However, more experiments are needed to investigate this issue.

Concerning memory usage, we found that, as expected, the smaller CHAR model required substantially smaller amounts of memory, especially during training as evidence by Figure 6. Note that, the RAM usage of the ORIGINAL model is mostly due to GloVe pre-trained vectors.

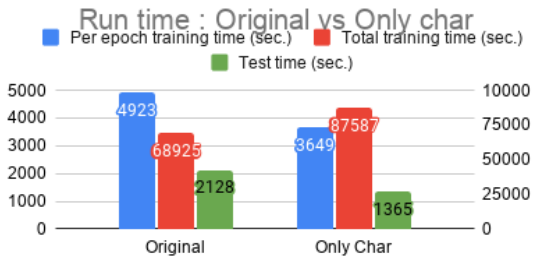

Figure 5: Run-time between the largest (54M weights) and smallest (677k weights) models. The total training time is associated with the right axis while the other measures are associated with the left axis.

5.2 Results 2: Static Embeddings

We now focus on the second set of experiments, focusing our attention to static embeddings. The models concerned are from Group 2 of Table 1.

First, we varied the static embedding (GloVe, Word2Vec, FastText), while keeping the same contextual embedding and character embedding as in the ORIGINAL model. It can be seen in Figure 7 that, when used with other embeddings (contextual and character), all static embeddings show comparable performance. The average performance ranging from 77.12 (GLOVE) to 75.59 (WORD2VEC). This corroborates with our earlier findings of section 5.1 whereby the model with only contextual and character embeddings, i.e. CONTEXTUAL/CHAR, achieved comparable performance to the ORIGINAL (static/contextual/char) model, indicating that the specific static embedding chosen contribute only marginally to the model's performance.

However, when used alone (see Figure 8), we see a drastic difference in performance between them; with the average performance ranging from 72.73 (GLOVE) to 51.56 (WORD2VEC).

Thus, it is only when studied alone that static embeddings show their differences. Once we iso-

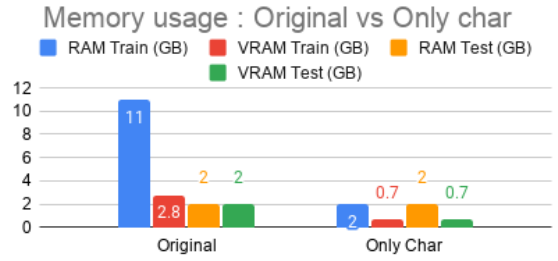

Figure 6: Memory usage between the largest (54M weights) and smallest (677k weights) models.

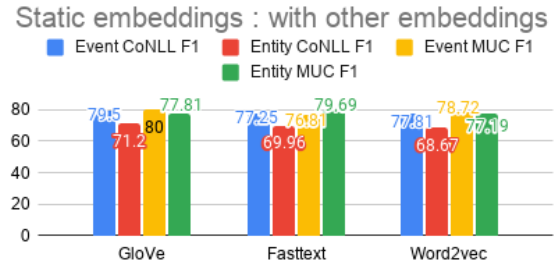

Figure 7: Comparing the predictive performance of static embeddings when used with other embeddings (ELMo and Character)

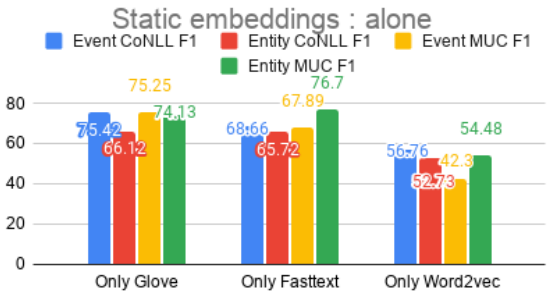

Figure 8: Comparing the predictive performance of static embeddings when used alone

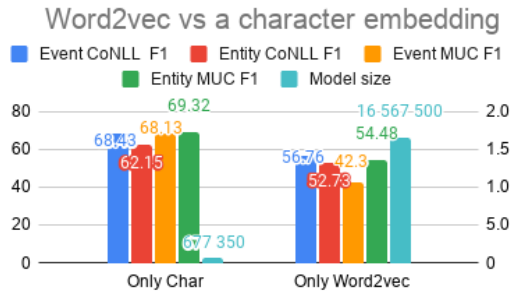

Figure 9: Comparing the predictive performance of solely Word2Vec vs solely a character embedding

late static embeddings, we see GloVe works best for EvCR. However, for EnCR, the FASTTEXT model show significantly higher MUC. The better performance of GloVe and FastText with respect to word2vec can be explained by their construction. Compared to Word2Vec, GloVe takes words co-occurrence information into account. If coreferent event mentions are more likely to share co-occurring words, it would explain parts of the performance gain. FastText also outperforms Word2Vec; here the difference is that FastText takes sub-word information into account which can be advantageous for coreferent entity mentions. E.g. in Figure 1, "Korea" and "Korean" have similar sub-word information.

What is most surprising is that Word2Vec is significantly outperformed by a simple character embedding as we can see on Figure 9. Moreover, in term of dimension Word2Vec has 300 and the character embedding has 50. Thus, the resulting model is not only more accurate but also ∼24 times smaller (Figure 9). This could indicate that the internal structure of a word (char embedding) contains more information about possible coreferences than its usual entourage (Word2Vec).

5.3 Results 3: Contextual Embeddings

We now focus on the third set of experiments about contextual embeddings. The models concerned are from Group 3 of Table 1.

Similarly to the previous section, we present the performance of different contextual embeddings when used in tandem with the static (GloVe) and character embedding of the original model (Figure 10) or when used alone (Figure 11). We see the same as in the previous section, i.e. the difference in performance between the contextual embeddings is clearer when they are used alone versus when they are used with GloVe and a character embedding. Thus, we will only focus on the Figure 11 which better represent the differences between ELMo, BERT, and GPT-2.

A first observation is that BERT both outperforms and is outperformed by GPT-2 on both tasks. Specifically, BERT performs better in EvCR while GPT-2 performs better in EnCR.

A second observation is that ELMo clearly outperforms GPT-2 and BERT on both tasks. This result contradicts Joshi et al. (2019) who found that BERT greatly outperforms ELMo on EnCR (+11.5 F1 on the GAP benchmark). Such disparity may be indicative of differences in the model and dataset.

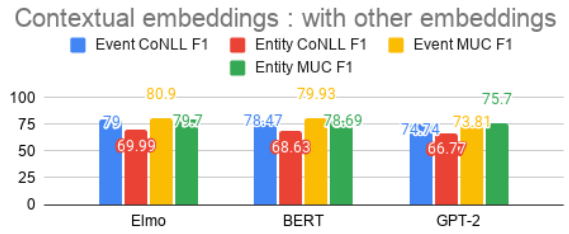

Figure 10: Comparing the predictive performance of contextual embeddings when used with other embeddings (GloVe and Character embedding)

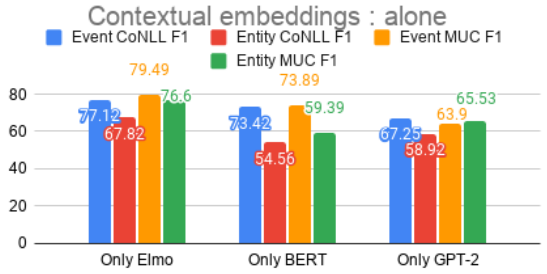

Figure 11: Comparing the predictive performance of contextual embeddings when used alone

Joshi et al. (2019) uses a span ranking approach which asks, for each mention, which is the most likely antecedent. This implicitly produces a tree which clusters coreferent mentions. Such method only takes local information between two mention into account while the method used in Barhom et al. (2019) uses global information between two entity clusters and related event clusters. Moreover, ECB+ or EventCorefBank+ is an EvCR dataset first and foremost and only defines EnCR to support EvCR; you could argue that the EnCR tasks is more about argument than entities. GAP on the other hand is a corpus of ambiguous pronoun-name pairs (Joshi et al., 2019).

Thus, while an EnCR task is defined by both dataset, they are significantly different. We argue that both the task definition and the use of global versus local information play a major role in the disparity between the performance reported by Joshi et al. (2019) and our study. Further confirming these findings would require evaluating Barhom et al. (2019)'s model on GAP and Joshi et al. (2019)'s on ECB+. However, these models are not interchangeable because the datasets and the task they define differs.

6 Conclusion

We used the state-of-the-art in EvCR (Barhom et al., 2019) as a framework to investigate the complexity-performance trade-off and compare the predictive performance of word embeddings across and within the three families.

We observed that the smallest model using solely a character embedding yielded 86% of the performance of the original (largest) model (using Elmo, GloVe and a character embeddings) despite being only 1.2% of its size. In fact, that smallest model achieves similar performance (68.43 F1) to the previous state-of-the-art in EvCR (69 F1) (Kenyon-Dean et al., 2018).

Paradoxically, we found that the largest model converged faster during training (by 21% in overall run-time) as it took only 14 epochs vs 24 for the character model. Overall, we found size and runtime to be weakly correlated.

In addition, our experiments revealed that augmenting the model with additional embeddings does not substantially improve the performance, leading to diminishing returns in term of predictive performance per embedding.

Concerning predictive performance, one of our most interesting result is that the model using solely a character embedding significantly outperformed (∼+10 F1) a larger model using solely a static embedding (Word2Vec) while being radically smaller (4% of its size). Hence, while character embeddings have often been used as supplementary embeddings, they can actually compete with other embeddings' families in terms of predictive performance per size.

Finally, our experiments lead us to conclude that for the task of Event and Entity Coreference Resolution, GloVe, FastText and Elmo yielded the best predictive performance. GloVe and FastText performed best in EvCR and EnCR respectively in their family while Elmo performs best overall.

Future directions include working on other comprehensive study of embeddings in other tasks and experimenting with CR models using different embeddings for different tasks to improve performance. E.g. GloVe and FastText in EvCR and EnCR respectively.

7 Ethical considerations

We trained 16 models over a two months period, estimated cost ranges from 350kWh to 400kWh. The estimated carbon impact ranges from 105Kg

to 120Kg of CO2 based on local data (300g CO2/kWh). We believe no other ethical considerations are raised by the content of this paper.

Acknowledgments

This research was funded by KPMG Belgium & Luxembourg through the HEC Digital Lab/HEC-Liège/ULiège.

References

Felipe Almeida and Geraldo Xexéo. 2019. Word embeddings: A survey. *CoRR*, abs/1901.09069.

Saliha Azzam, Kevin Humphreys, and Robert Gaizauskas. 1999. Using coreference chains for text summarization. In *Coreference and Its Applications*.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Cosmin Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1412–1422, Uppsala, Sweden. Association for Computational Linguistics.

Giacomo Berardi, Andrea Esuli, and Diego Marcheggiani. 2015. Word embeddings go to italy: A comparison of models and training datasets. In *IIR*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Juntian Chen, Yubo Tao, and Hai Lin. 2018. Visual exploration and comparison of word embeddings. *Journal of Visual Languages & Computing*, 48:178 – 186.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133, Copenhagen, Denmark. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552, Reykjavik, Iceland. European Language Resources Association (ELRA).

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. Association for Computational Linguistics.

Cícero Nogueira dos Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1818–1826. JMLR.org.

Dagmar Gromann and Thierry Declerck. 2018. Comparing pretrained multilingual word embeddings on an ontology alignment task. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Kevin Humphreys, Robert Gaizauskas, and Saliha Azzam. 1997. Event coreference for information extraction. In *Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*.

Mandar Joshi, Omer Levy, Luke Zettlemoyer, and Daniel Weld. 2019. BERT for coreference resolution: Baselines and analysis. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5803–5808, Hong Kong, China. Association for Computational Linguistics.

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 1–10, New Orleans, Louisiana. Association for Computational Linguistics.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500, Jeju Island, Korea. Association for Computational Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Hongmin Li, Xukun Li, Doina Caragea, and Cornelia Caragea. 2018. Comparison of word embeddings and sentence encodings as generalized representations for crisis tweet classification tasks. *Proceedings of ISCRAM Asia Pacific*.

Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joseph E. Gonzalez. 2020. Train large, then compress: Rethinking model size for efficient training and inference of transformers.

Qi Liu, Matt J. Kusner, and Phil Blunsom. 2020. A survey on contextual embeddings.

Jing Lu and Vincent Ng. 2018. Event coreference resolution: A survey of two decades of research. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 5479–5486. ijcai.org.

Lesly Miculicich Werlen and Andrei Popescu-Belis. 2017. Using coreference links to improve Spanish-to-English machine translation. In *Proceedings of the 2nd Workshop on Coreference Resolution Beyond OntoNotes (CORBON 2017)*, pages 30–40, Valencia, Spain. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Nafise Sadat Moosavi and Michael Strube. 2016. Which coreference evaluation metric do you trust? a proposal for a link-based entity aware metric. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 632–642, Berlin, Germany. Association for Computational Linguistics.

Jeffrey Pennington, 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)*, pages 1532–1543, Doha, Qatar. 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. 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, Jeju Island, Korea. Association for Computational Linguistics.

Nikolaos Stylianou and Ioannis Vlahavas. 2021. A neural entity coreference resolution review. *Expert Systems with Applications*, 168:114466.

Jian Su, Xiaofeng Yang, Huaqing Hong, Yuka Tateisi, and Jun'ichi Tsujii. 2008. Coreference Resolution in Biomedical Texts: a Machine Learning Approach. In *Ontologies and Text Mining for Life Sciences : Current Status and Future Perspectives*, number 08131 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany.

Rhea Sukthanker, Soujanya Poria, Erik Cambria, and Ramkumar Thirunavukarasu. 2018. Anaphora and coreference resolution: A review.

Luchen Tan, Haotian Zhang, Charles Clarke, and Mark Smucker. 2015. Lexical comparison between Wikipedia and Twitter corpora by using word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 657–661, Beijing, China. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Florence, Italy. Association for Computational Linguistics.

Yanshan Wang, Sijia Liu, Naveed Afzal, Majid Rastegar-Mojarad, Liwei Wang, Feichen Shen, Paul Kingsbury, and Hongfang Liu. 2018. A comparison of word embeddings for the biomedical natural language processing. *Journal of Biomedical Informatics*, 87:12 – 20.