A Multi-Strategy Approach to Crossword Clue Answer Retrieval and Ranking

Andrea Zugarini[1,2], **Marco Ernandes**[1]
1. Expert.ai, Italy
2. DIISM University of Siena, Italy
{azugarini,mernandes}@expert.ai

Abstract

English. Crossword clues represent an extremely challenging form of Question Answering, due to their intentional ambiguity. Databases of previously answered clues are a vital source for the retrieval of candidate answers lists in Automatic Crossword Puzzles (CPs) resolution systems. In this paper, we exploit language neural representations for the retrieval and ranking of crossword clues and answers. We assess the performances of several embedding models, both static and contextual, on Italian and English CPs. Results indicate that embeddings usually outperform the baseline. Moreover, the use of embeddings for retrieval allows different ranking strategies, which turned out to be complementary, and lead to better results when used in combination.

Italiano. *Le domande dei cruciverba rappresentano una forma di Question Answering particolarmente complessa a causa della loro intenzionale ambiguità. I risolutori automatici di cruciverba sfruttano ampiamente basi di dati di domande precedentemente risposte. In questo articolo proponiamo l'uso di embeddings per la ricerca semantica di domande-risposte da tali databases. Le performances sono valutate in cruciverba di lingua sia italiana che inglese, confrontando diversi tipi di embeddings, sia contestuali che statici. I risultati suggeriscono che la ricerca semantica è migliore della baseline. Inoltre, l'utilizzo di embeddings permette di applicare differenti strategie di retrieval, che, migliorano la qualità dei risultati quando usate congiuntamente.*

1 Introduction

Crossword Puzzles (CPs) resolution is a popular game. As almost any other human game, it is possible to tackle the problem automatically. CPs solvers frame it into a constraint satisfaction task, where the goal is to maximize the probability of filling the grid with answers consistent with their clues and coherent to the puzzle scheme. These systems (Littman et al., 2002; Ernandes et al., 2005; Ginsberg, 2011) heavily rely on lists of candidate answers for each clue. Candidates' quality is crucial to CPs resolution. If the correct answer is not present in the candidates' list, the Crossword Puzzle cannot be solved correctly. Moreover, even a poorly ranked correct answer can lead to a failure in the crossword puzzle filling. Answers lists can come from multiple solvers, where each solver is typically specialized in solving different kinds of clues, and/or exploits different source of information. Such lists are mainly retrieved with two techniques: (1) by querying the web with search engines using clue representations; (2) interrogating clue-answer databases that contain previously answered clues. In this work, we focus on the latter.

In the problem of candidate answers retrieval from clue-answer knowledge sources, answers are ranked according to the similarity between a query clue and the clues in the DB. The similarity is provided by the search engine that assigns a score to each retrieved answer. Several approaches have been carried out to re-rank the candidates' list by means of learning to rank strategies (Barlacchi et al., 2014a; Barlacchi et al., 2014b; Nicosia et al., 2015; Nicosia and Moschitti, 2016; Severyn et al., 2015). These approaches require a training phase to learn how to rank and mostly differ for the re-

ranking model or strategy adopted. In particular, pre-trained distributed representations and neural networks are used for re-ranking clues in (Severyn et al., 2015).

The re-ranking of answer candidates attempts to improve the quality of candidates' lists, assuming that the correct answer belongs to the list. Differently from previous work, we aim at directly retrieving richer lists of answer candidates from a clue-answer database. In order to do so, we exploit both static and contextual distributed representations to perform a semantic search on the DB. An embedding-based search extends the retrieval to semantically related clues that may be phrased differently. Moreover, it also allows us to map in the same space questions and answers, which opens the way for ranking answers directly based on their similarity with respect to the query clue. Our approach requires no training on CPs data and it can be applied with any pre-trained embedding model.

In summary, the contributions of this work are: (1) a semantic search approach to candidate answer retrieval in automatic crossword resolution; (2) two complementary retrieval methodologies (namely QC and QA) detecting candidate answers that when combined together (even naively) produce a better set of candidates; (3) a comparison between different pre-trained language representations (either static or contextual).

The paper is organized as follows. First, we describe in Section 2 distributed representations of language. In Section 3, we present the two answer retrieval approaches proposed in this work. Then, in Section 4 we outline the experiments in detail, and discuss the obtained results. Finally, we draw our conclusions in Section 5.

2 Language Representations

Assigning meaningful representations to language is a long standing problem. Since the inception of the first text mining solutions, the bag-of-words technique has been widely adopted as one of the standard approaches to text representation. Inverted indices and statistical weighting schemes (as TF-IDF or BM25) are still to this day commonly paired with bag-of-words, providing a scalable and effective approach to document retrieval. On the other hand, in the last decade, we have assisted to tremendous progress in the field of Natural Language Processing. Huge credit goes to the diffusion of distributed representations of words (Bengio et al., 2003; Mikolov et al., 2013a; Mikolov et al., 2013b; Collobert et al., 2011; Mikolov et al., 2018; Devlin et al., 2018) learned through Language Modeling related tasks on large corpora.

In general, the goal is to assign a fixed length representation of size d, aka embedding, to a textual passage s such that similar text passages - syntactically and/or semantically - are represented closely in such space. An embedding model f_e is a function mapping s to a d-dimensional vector, i.e: $f_e : s \rightarrow \mathbb{R}^d$. Since language is a composition of symbols (typically words), embedding models first tokenize text and then process such tokens in order to compute the representation of such textual passage.

Nowadays, there are lots of embedding models, and for some of them pre-trained embeddings are available in a plethora of languages (Yamada et al., 2020; Grave et al., 2018; Yang et al., 2019). Early methods like (Mikolov et al., 2013a) produce dense representations for single tokens - mainly words - therefore further processing is needed to obtain the actual representation of s, when s is composed of multiple words. These kinds of embeddings are also referred to as static embeddings, since the representation of a token is always the same regardless of the context in which it appears. In (Mikolov et al., 2018), authors extend (Mikolov et al., 2013a) introducing n-gram and sub-word information and in (Le and Mikolov, 2014), distributed representations are learned directly for sentences and documents.

Most of the proposed methods for contextual embeddings were based on recurrent neural language models (Melamud et al., 2016; Yang et al., 2019; Chidambaram et al., 2018; Mikolov et al., 2010; Marra et al., 2018; Peters et al., 2018), until the introduction of transformer architectures (Vaswani et al., 2017; Devlin et al., 2018; Liu et al., 2019) which are currently the state-of-the-art models. In the next Section we will discuss how such representations can be used to perform semantic search. In the experiments, we will exploit some of these embedding models - both static and contextual.

3 Semantic Search

Traditional CPs solvers rely on Similar Clue Retrieval mechanisms. The idea is to find possible

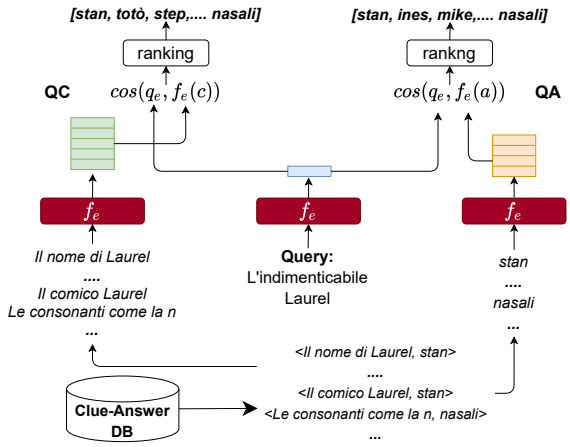

Figure 1: Sketch of the two answer candidates retrieval approaches: QC (on the left), QA (on the right). In QC, ranking is based on the similarity between the query embedding and all the clues in the DB, while in QA the similarity is computed between the query and the answers in the database.

answers from clues in the database that are similar to the given query. This is particularly effective for crosswords, since the same clues tend to be repeated over time, or may have little lexical variations. Retrieval of similar clues is based on search engines based on classical IR algorithms such as TF-IDF or BM25, representing clues in the database as documents to retrieve, given the target clue as query.

Here instead, we retrieve and rank documents with semantic search. We propose two strategies, namely QC and QA. QC is analogous to classical similar clues retrieval systems, with the difference that text is represented with a dense representation. The approach retrieves and ranks from the DB clues similar to the query and returns in output the answers associated to those clues. QA, instead, ranks the answers directly by computing the cosine distance between the query and the answers themselves. Intuitively, the latter approach ranks well answers semantically correlated to the question itself, particularly useful for clues about synonyms. As we will show in Section 4, due to their different nature, the list of candidates retrieved by the two approaches are strongly complementary. A sketch of the two approaches is outlined in Fig. 1. Let us describe them separately.

3.1 Similar Clues Retrieval

We are given a query clue which is a sequence of n words $q := (w_1, \ldots, w_n)$, and a clue-answer DB (C, A) constituted by M clue-answer pairs, where C and A indicate the list of all the clues and answers, respectively, while we denote a clue-answer pair as: (c, a).

We assign a fixed-length representation $q_e \in \mathbb{R}^d$ to the query clue q, computed with an embedding model:

$$q_e = f_e(q). \tag{1}$$

For contextual embeddings f_e is the model itself, since they work directly on the sequence, whereas for static embeddings we have to collapse n word representations together into a single vector. For simplicity, we simply average such embeddings.

Analogously, each clue $c \in C$ is encoded as in Equation 1. Then, we measure the cosine similarity between the query and each clue:

$$score(q, (c, a)) = cos(q_e, f_e(c)), \tag{2}$$

where $cos(\cdot, \cdot)$ denotes the cosine similarity. Thus, we obtain a similarity score for each clue-answer pair. In order to finally rank answers we average all clue-answer pairs having the same answer:

$$score(q, a) = \frac{1}{|\mathcal{A}|} \cdot \sum_{a_k \in \mathcal{A}} score(q, (c, a_k)), \tag{3}$$

where $\mathcal{A}$ indicates the set of clue-answer pairs where the answer a_k is equal to a. All the answers in A are then ranked. Since we know a priori the length of a query answer, candidates with incorrect lengths are filtered out. We refer to this approach as QC (Query-Clue).

3.2 Similar Answers Retrieval

Since we can map text into a fixed-length space, we can also rank by measuring the similarity between the query and the answer itself. The query is encoded exactly as in Equation 1. In this case however we only need the clue-answer DB to retrieve the set of unique answers, denoted as $\mathbf{A}$. Similarly to Equation 2, we compute the cosine similarity between query and answer embeddings:

$$score(q, a) = cos(q_e, f_e(a)), \tag{4}$$

for each $a \in \mathbf{A}$, then we rank as in QC. We call it QA (Query-Answer). It is important to remark that QA is only feasible using latent representations, traditional methods like TF-IDF are not suited because of their sparsity of representations. Moreover, QA is somewhat an orthogonal strategy with respect to QC. We will see in Section 4, how even a trivial ensemble of QA and QC is beneficial to the performances.

4 Experiments

In the experiments we aim to prove the effectiveness of semantic search to retrieve accurate lists of candidate answers, and to show that the QA approach carries out complementary information that can increase the coverage of the retrieval.

4.1 Experimental Setup

We considered for our experiments three well known embedding models, two static (Word2Vec[1][2], FastText[3]) and one contextual (Universal Sentence Encoder[4]), briefly denoted as W2V, FT and USE, respectively. We exploited pre-trained models for all of them. In absence of an Italian USE model, we used for the Italian crosswords database, the multilingual version of USE, that was trained on 16 languages (Italian included). Embedding models are compared against TF-IDF, which is a typical text representation in document retrieval problems.

To measure performances, we used well known metrics of Retrieval systems. In particular we considered Mean Hit at k (MH@k) and Mean Reciprocal Rank (MRR). Hit at k is 1 if the correct answer is within the first k elements of the list, 0 otherwise. The hits at k are evaluated for $k = \{1, 5, 20, 100\}$. MRR is defined as follows: $\frac{1}{n}\sum_{q=1}^{n}\frac{1}{rank(q)}$.

4.2 Datasets

We consider two different clue-answer databases for our experimentation. In particular, experiments were carried out on two languages, Italian and English, respectively on CWDB dataset (Barlacchi et al., 2014a) and New York Times Crosswords. We apply the same pre-processing pipeline in both corpora. (1) We discarded clue-answer pairs having answers with more than three characters, because they are typically about linguistic puzzles and they are addressed differently in CPs solvers. (2) Answer and clues containing special characters are erased. (3) Text has been lowercased and punctuation removed. (4) We kept only answers appearing in at least two clues.

[1]English: https://code.google.com/archive/p/word2vec/

[2]Italian: https://wikipedia2vec.github.io/wikipedia2vec/

[3]https://fasttext.cc/

[4]https://tfhub.dev/google/collections/universal-sentence-encoder/1

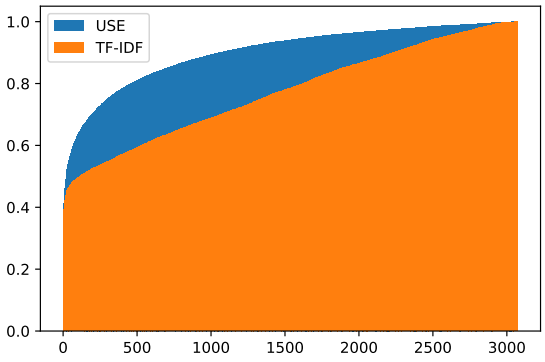

Figure 2: Comparison between cumulative density functions of ranking using USE (blue) and TF-IDF (orange) methods on English crosswords.

English Crosswords. The data consist of a collection of clue-answer pairs for crossword puzzles published in the New York Times[5] in 1997 and 2005, previously collected in (Ernandes et al., 2008). Overall, there are about $61,000$ clue-answer pair samples. Clues, answers and clue-answer pairs may occur multiple times. A clue is generally a short sentence, while answers are usually made up of a single word, but there are cases of multi-word answers. In such a case the answer is a string made of multiple words without any word separator. After pre-processing we obtain a corpus with $31,808$ pairs in which $27,527$ questions and $8,324$ answers are unique.

Italian Crosswords. The clue-answer database for Italian was constructed from *CWDB v0.1 it* corpus[6] (Barlacchi et al., 2014a). We combined pairs from both train and test splits, since we did not perform any training in our experiments and we opportunely omitted the clue-answer pair itself during its evaluation. From the original $62,011$ pairs, it remains $25,545$ pairs after pre-processing, constituted of $5,813$ unique answers and $16,970$ unique questions.

4.3 Results

All the results for Italian and English crosswords are outlined in Tables 1 and 2, respectively. From them, we can catch several interesting insights. First of all, contextual representations from Universal Sentence Encoders are gen-

[5]https://www.nytimes.com/

[6]https://ikernels-portal.disi.unitn.it/projects/webcrow

Model	Strategy	MH@1	MH@5	MH@20	MH@100	MRR
W2V	QA	**14.97**	**32.55**	**50.35**	**71.59**	**23.80**
FT	QA	6.78	14.47	26.88	52.46	11.44
USE	QA	7.89	17.81	29.30	46.80	13.24
TF-IDF	QC	**60.79**	66.43	68.53	72.62	63.54
W2V	QC	52.34	64.75	72.58	82.66	58.26
FT	QC	23.50	34.13	45.94	64.09	29.05
USE	QC	60.69	**70.93**	**76.81**	**84.70**	**65.57**
Ensemble$_{USE-W2V}$	QC-QA	-	**73.59**	**82.39**	**91.22**	-

Table 1: Evaluation of performances on CWDB Italian data. The best values of each column and strategy are marked in bold for both QC and QA methods.

Model	Strategy	MH@1	MH@5	MH@20	MH@100	MRR
W2V	QA	7.58	17.27	27.78	42.62	12.66
FT	QA	7.72	17.35	27.29	43.42	12.75
USE	QA	**8.63**	**19.69**	**30.01**	**45.17**	**14.25**
TF-IDF	QC	**26.15**	37.62	44.09	49.54	31.46
W2V	QC	19.63	31.69	42.66	57.38	25.65
FT	QC	15.72	24.32	32.67	46.64	20.20
USE	QC	25.78	**38.57**	**49.34**	**63.35**	**32.12**
Ensemble$_{USE-USE}$	QC-QA	-	**41.40**	**54.34**	**69.00**	-

Table 2: Evaluation of performances on English data. The best values of each column and strategy are marked in bold for both QC and QA methods.

erally the most effective ones, especially on similar clues retrieval (QC), where both the query and the elements to rank are textual sequences. Nonetheless, Word2vec embeddings work surprisingly well, outperforming FastText almost all the times. Furthermore, they are the best ones on QA search in Italian database. We believe the reason why Word2Vec outperforms USE on Italian QA is twofold. First, the advantage of contextual embeddings is less evident in QA setup, indeed USE brings less benefits on English QA as well. Second, USE is a multilingual model, therefore its embeddings are less specialized than Word2Vec which was instead trained for Italian only.

When comparing semantic search models against the baseline (TF-IDF) - which is only possible in QC - we can notice that, static embeddings struggle to outperform it. Indeed, the sparse nature of TF-IDF induces crisp similarity scores, very high for clues sharing the same keywords, extremely low for all the rest. On the contrary, similarity scores are more blurred with dense embeddings. As a consequence, TF-IDF achieves high MH@1 and MH@5 scores (and MRR too). However, TF-IDF leads to a poorer coverage when the candidates list grows (MH@20 and MH@100). This behavior is also evident in Fig. 2, where we compare the cumulative distributions of ranking with USE and TF-IDF. After the initial bump, TF-IDF hits growth is almost linear (i.e. random), whereas the Universal Sentence Encoder keeps growing significantly.

Ensembling QC and QA. Analyzing the results, we observed that ranks from QA and QC had low levels of overlaps. We reported in the last line of Tables 1 and 2, performances of a naive ensemble approach to combine QC and QA strategies. Due to the limited levels of overlaps, we decided to merge the two ranks taking the first $K/2$ ranks from each strategy to compute MH@K, $K = \{5, 20, 100\}$[7]. We chose the best embedding model on each strategy. Despite its simplicity and the large room for improvements, the ensemble significantly improved the performances in both languages. This suggests possible directions for further improving the retrieval of CPs solvers.

[7]Since K=5 is not even, we took the first 3 ranks from QC and the first two ranks from QA.

5 Conclusions

In this paper, we proposed two different semantic search strategies (QC and QA) for ranking and retrieving answer candidates to CPs clues. We exploited pre-trained state-of-the-art embeddings, both static and contextual, to rank clue-answer pairs from databases. Embedding-based retrieval overcomes some of the limitations of inverted indices models, leading to higher coverage ranks, and allowing similar answers retrieval (QA). Finally, we observed that, even a simple ensembling that combines QC and QA, is effective and improves overall retrieval performances.

This opens further research directions, where learning to rank methods could be exploited in order to better combine candidate answer lists from complementary approaches like QC and QA.

Acknowledgments

We thank Nicola Landolfi and Marco Maggini for the great support and fruitful discussions.

References

Gianni Barlacchi, Massimo Nicosia, and Alessandro Moschitti. 2014a. Learning to rank answer candidates for automatic resolution of crossword puzzles. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 39–48.

Gianni Barlacchi, Massimo Nicosia, and Alessandro Moschitti. 2014b. A retrieval model for automatic resolution of crossword puzzles in italian language. In *The First Italian Conference on Computational Linguistics CLiC-it 2014*, page 33.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Muthuraman Chidambaram, Yinfei Yang, Daniel Cer, Steve Yuan, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Learning cross-lingual sentence representations via a multi-task dual-encoder model. *arXiv preprint arXiv:1810.12836*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Marco Ernandes, Giovanni Angelini, and Marco Gori. 2005. Webcrow: A web-based system for crossword solving. In *AAAI*, pages 1412–1417.

Marco Ernandes, Giovanni Angelini, and Marco Gori. 2008. A web-based agent challenges human experts on crosswords. *AI Magazine*, 29(1):77–77.

Matthew L Ginsberg. 2011. Dr. fill: Crosswords and an implemented solver for singly weighted csps. *Journal of Artificial Intelligence Research*, 42:851–886.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. 2018. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196. PMLR.

Michael L Littman, Greg A Keim, and Noam Shazeer. 2002. A probabilistic approach to solving crossword puzzles. *Artificial Intelligence*, 134(1-2):23–55.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Giuseppe Marra, Andrea Zugarini, Stefano Melacci, and Marco Maggini. 2018. An unsupervised character-aware neural approach to word and context representation learning. In *International Conference on Artificial Neural Networks*, pages 126–136. Springer.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 51–61.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Massimo Nicosia and Alessandro Moschitti. 2016. Crossword puzzle resolution in italian using distributional models for clue similarity. In *IIR*.

Massimo Nicosia, Gianni Barlacchi, and Alessandro Moschitti. 2015. Learning to rank aggregated answers for crossword puzzles. In *European Conference on Information Retrieval*, pages 556–561. Springer.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Aliaksei Severyn, Massimo Nicosia, Gianni Barlacchi, and Alessandro Moschitti. 2015. Distributional neural networks for automatic resolution of crossword puzzles. 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 199–204.

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.

Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. 2020. Wikipedia2Vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from Wikipedia. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 23–30. Association for Computational Linguistics.

Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. 2019. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*.