Distilling Relation Embeddings from Pre-trained Language Models

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

Asahi Ushio
Jose Camacho-Collados
Steven Schockaert

https://github.com/asahi417/relbert

Language Model Understanding

Syntactic Knowledge

- Probing embedding: Hewitt 2019, Tenney 2019
- Probing attention weight: Clark 2020

Factual Knowledge a.k.a Language Model as a Commonsense KB

- Petroni 2019, Kassner 2020, Jiang 2020, etc

Relational Knowledge a.k.a Language Model as a Lexical Relation Reasoner

- LM fine-tuning on relation classification: Bouraoui 2019
- Vanilla LM evaluation: Ushio 2021

Language Model Understanding

Syntactic Knowledge

- Probing embedding: Hewitt 2019, Tenney 2019
- Probing attention weight: Clark 2020

Factual Knowledge a.k.a Language Model as a Commonsense KB

- Petroni 2019, Kassner 2020, Jiang 2020, etc

Relational Knowledge a.k.a Language Model as a Lexical Relation Reasoner

- LM fine-tuning on relation classification: Bouraoui 2019
- Vanilla LM evaluation: Ushio 2021

Language Model Understanding

Syntactic Knowledge

- Probing embedding: Hewitt 2019, Tenney 2019
- Probing attention weight: Clark 2020

Factual Knowledge a.k.a Language Model as a Commonsense KB

- Petroni 2019, Kassner 2020, Jiang 2020, etc

Relational Knowledge a.k.a Language Model as a Lexical Relation Reasoner

- LM fine-tuning on relation classification: Bouraoui 2019
- Vanilla LM evaluation: Ushio 2021

Can we distil relational knowledge as relation embedding?

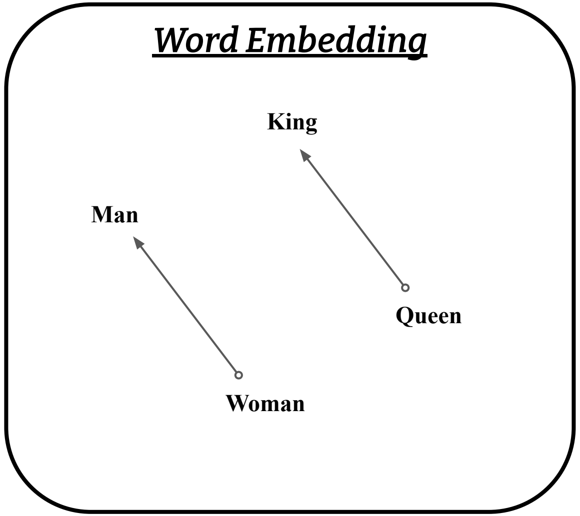
Relation Embedding

Word Embedding [Mikolov (2013)]

Pair2Vec [Joshi (2019)]

Relative [Camacho-Collados (2019)]

X	Y	Contexts
hot	*cold*	with **X** and **Y** baths too **X** or too **Y** neither **X** nor **Y**
Portland	*Oregon*	in **X**, **Y** the **X** metropolitan area in **Y** **X** International Airport in **Y**
crop	*wheat*	food **X** are maize, **Y**, etc dry **X**, such as **Y**, more **X** circles appeared in **Y** fields
Android	*Google*	**X** OS comes with **Y** play the **X** team at **Y** **X** is developed by **Y**



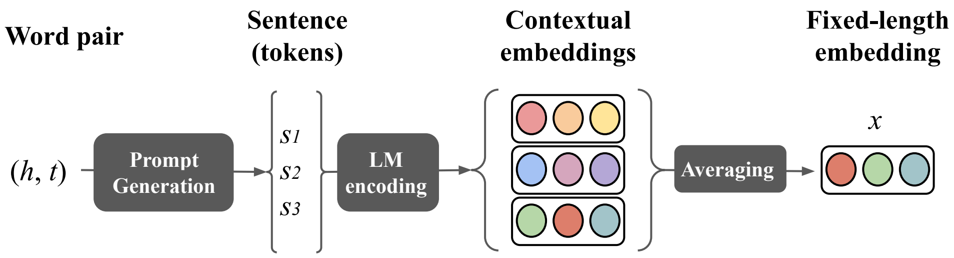
RelBERT

Relation Embedding from LM

Prompt Generation
Custom Template, AutoPrompt (Shin 2020), P-tuning (Liu 2021)

LM embedding

Averaging over the context



Relation Embedding from LM

Prompt Generation

Custom Template, AutoPrompt (Shin 2020), P-tu

1. Today, I finally discovered the relation between **camera** and **photographer** : **camera** is the <mask> of **photographer**

2. Today, I finally discovered the relation between **camera** and **photographer** : **photographer** is **camera**'s <mask>

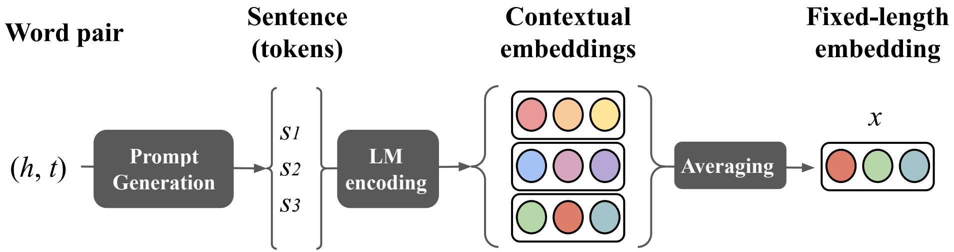
3. Today, I finally discovered the relation between **camera** and **photographer** : <mask>

4. I wasn't aware of this relationship, but I just read in the encyclopedia that **camera** is the <mask> of **photographer**

5. I wasn't aware of this relationship, but I just read in the encyclopedia that **photographer** is **camera**'s <mask>

LM embedding

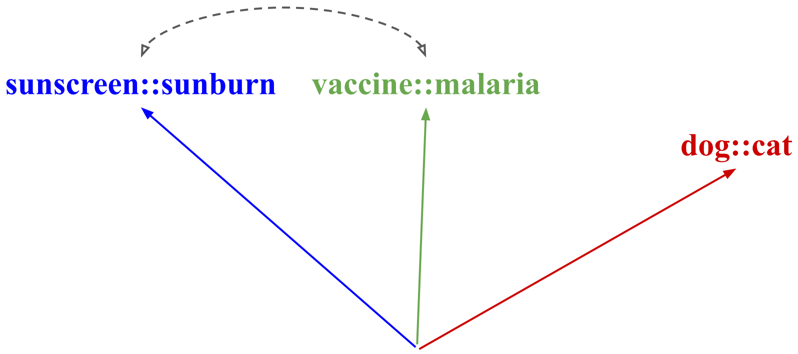
Averaging over the context



Fine-tuning on Triples

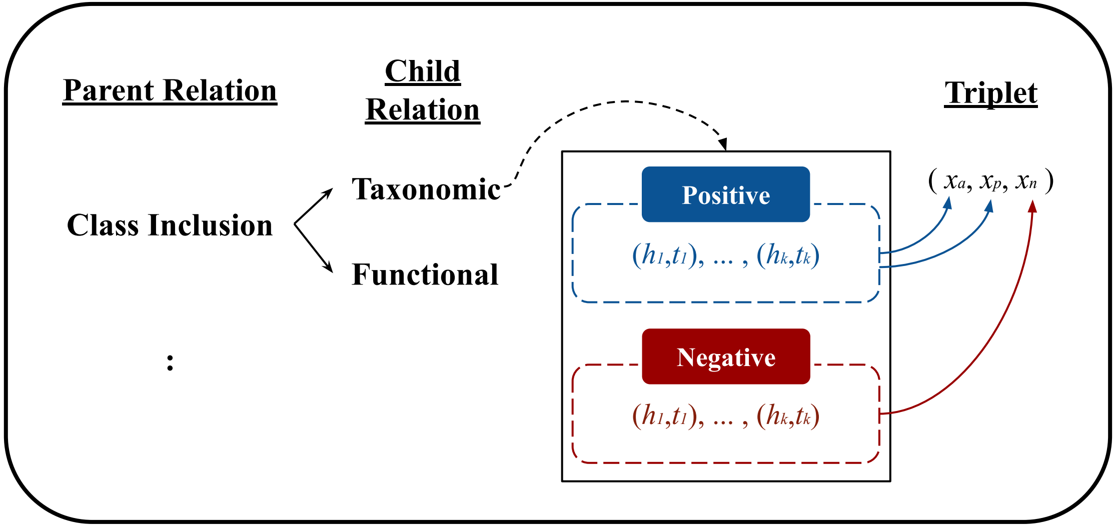
Given a triple: anchor "`sunscreen::sunburn`", positive "`vaccine::malaria`", and negative "`dog::cat`", we want the embeddings of the anchor and the posirive close but far from the negative.

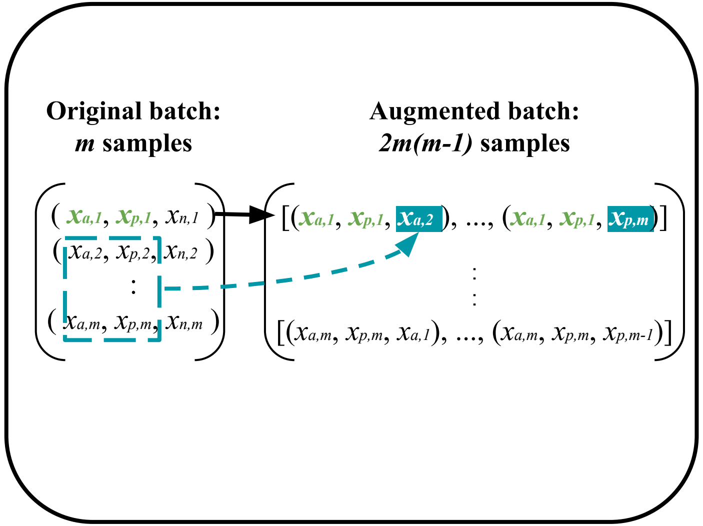
Loss function: *Triplet loss* and *classification loss* following SBERT (Reimers 2019).



Dataset

We create the dataset from **SemEval 2012 Task 2**.





EXPERIMENTS

Experiment: Analogy

Query:		word:language
Candidates:	(1)	paint:portrait
	(2)	poetry:rhythm
	(3)	**note:music**
	(4)	tale:story
	(5)	week:year

Sample from SAT analogy dataset.

Dataset	Data size (val / test)	No. candidates	No. groups
SAT	37 / 337	5	2
UNIT 2	24 / 228	5,4,3	9
UNIT 4	48 / 432	5,4,3	5
Google	50 / 500	4	2
BATS	199 / 1799	4	3

Data statistics.

Setup

- Cosine similarity in between embeddings.
- No training.
- Accuracy as the metric.
- No validation.

Experiment: Analogy

SotA in 4 / 5 datasets 🎉

Better than tuned methods on dev set 🤗

Model	SAT†	SAT	U2	U4	Google	BATS
GPT-3 (zero)	*53.7*	-	-	-	-	-
GPT-3 (few)	*65.2**	-	-	-	-	-
RELATIVE	24.9	24.6	32.5	27.1	62.0	39.0
pair2vec	33.7	34.1	25.4	28.2	66.6	53.8
FastText	49.7	47.8	43.0	40.7	**96.6**	72.0
Analogical Proportion Score (tuned)						
· GPT-2	*57.8**	*56.7**	*50.9**	*49.5**	*95.2**	*81.2**
· BERT	*42.8**	*41.8**	*44.7**	*41.2**	*88.8**	*67.9**
· RoBERTa	*55.8**	*53.4**	*58.3**	*57.4**	*93.6**	*78.4**
RelBERT						
· Manual	**69.5**	**70.6**	**66.2**	**65.3**	92.4	**78.8**
· AutoPrompt	61.0	62.3	61.4	63.0	88.2	74.6
· P-tuning	54.0	55.5	58.3	55.8	83.4	72.1

Experiment: Classification

Setup

- Supervised Task
- LMs are frozen
- macro/micro F1
- Tuned on dev

	BLESS	CogALex	EVALution	K&H+N	ROOT09
Random	8,529/609/3,008	2,228/3,059	-	18,319/1,313/6,746	4,479/327/1,566
Meronym	2,051/146/746	163/224	218/13/86	755/48/240	-
Event	2,657/212/955	-	-	-	-
Hypernym	924/63/350	255/382	1,327/94/459	3,048/202/1,042	2,232/149/809
Co-hyponym	2,529/154/882	-	-	18,134/1,313/6,349	2,222/162/816
Attribute	1,892/143/696	-	903/72/322	-	-
Possession	-	-	377/25/142	-	-
Antonym	-	241/360	1,095/90/415	-	-
Synonym	-	167/235	759/50/277	-	-

Data statistics.

Experiment: Classification

SotA in 4 / 5 datasets in macro F1 score 🎉

SotA in 3 / 5 datasets in micro F1 score 🎉

Model		BLESS		CogALexV		EVALution		K&H+N		ROOT09	
		macro	micro	macro	micro	macro	micro	macro	micro	macro	micro
GloVe	*cat*	92.9	93.3	42.8	73.5	56.9	58.3	88.8	94.9	86.3	86.5
	cat+dot	**93.1**	93.7	51.9	79.2	55.9	57.3	89.6	95.1	88.8	89.0
	cat+dot+pair	91.8	92.6	56.4	81.1	58.1	59.6	89.4	95.7	89.2	89.4
	cat+dot+rel	91.1	92.0	53.2	79.2	58.4	58.6	89.3	94.9	89.3	89.4
	diff	91.0	91.5	39.2	70.8	55.6	56.9	87.0	94.4	85.9	86.3
	diff+dot	92.3	92.9	50.6	78.5	56.5	57.9	88.3	94.8	88.6	88.9
	diff+dot+pair	91.3	92.2	55.5	80.2	56.0	57.4	88.0	95.5	89.1	89.4
	diff+dot+rel	91.1	91.8	52.8	78.6	56.9	57.9	87.4	94.6	87.7	88.1
FastText	*cat*	92.4	92.9	40.7	72.4	56.4	57.9	88.1	93.8	85.7	85.5
	cat+dot	92.7	93.2	48.5	77.4	56.7	57.8	89.1	94.0	88.2	88.5
	cat+dot+pair	90.9	91.5	53.0	79.3	56.1	58.2	88.3	94.3	87.7	87.8
	cat+dot+rel	91.4	91.9	50.6	76.8	57.9	59.1	86.9	93.5	87.1	87.4
	diff	90.7	91.2	39.7	70.2	53.2	55.5	85.8	93.3	85.5	86.0
	diff+dot	92.3	92.9	49.1	77.8	55.2	57.4	86.5	93.6	88.5	88.9
	diff+dot+pair	90.0	90.8	53.9	79.0	55.8	57.8	86.6	94.2	87.7	88.1
	diff+dot+rel	90.6	91.3	53.6	78.2	57.1	58.0	86.3	93.4	86.9	87.4
RelBERT	Manual	91.7	92.1	**71.2**	**87.0**	68.4	69.6	88.0	96.2	**90.9**	**91.0**
	AutoPrompt	91.9	92.4	68.5	85.1	**69.5**	**70.5**	**91.3**	97.1	90.0	90.3
	P-tuning	91.3	91.8	67.8	84.9	69.1	70.2	88.5	96.3	89.8	89.9
SotA	LexNET	-	89.3	-	-	-	60.0	-	98.5	-	81.3
	SphereRE	-	**93.8**	-	-	-	62.0	-	**99.0**	-	86.1

ANALYSIS

Relation Memorarization

Does RelBERT just memorize the relations in the training set... ?

Experiment: Train RelBERT without hypernymy.

Result: No significant decrease in hyperbyny prediction.

→ RelBERT **does not** rely on the memorization!

	BLESS	CogALex	EVAL	K&H+N	ROOT09
rand	93.7 (+0.3)	94.3 (-0.2)	-	97.9 (+0.2)	91.2 (-0.1)
mero	89.8 (+1.4)	72.9 (+2.7)	69.2 (+1.9)	74.5 (+5.4)	-
event	86.5 (-0.3)	-	-	-	-
hyp	94.1 (+0.8)	60.9 (-0.7)	61.7 (-1.5)	93.5 (+5.0)	83.0 (-0.4)
cohyp	96.4 (+0.3)	-	-	97.8 (+1.2)	97.4 (-0.5)
attr	92.6 (+0.3)	-	84.7 (+1.6)	-	-
poss	-	-	67.1 (-0.2)	-	-
ant	-	76.8 (-2.6)	81.3 (-0.9)	-	-
syn	-	49.9 (-0.6)	53.6 (+2.7)	-	-
macro	92.2 (+0.5)	71.0 (-0.2)	69.3 (+0.9)	90.9 (+2.9)	90.5 (-0.4)
micro	92.5 (+0.4)	86.9 (-0.1)	70.2 (+0.6)	97.2 (+1.0)	90.7 (-0.3)

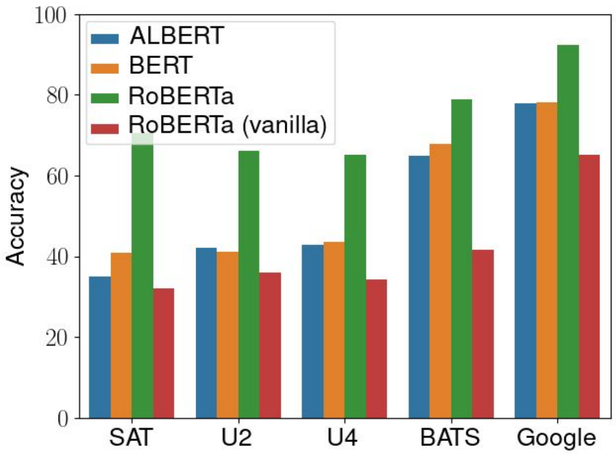
Fine-tuning? Other LMs?

Train RelBERT on BERT, ALBERT in addition to RoBERTa.

→ **RoBERTa is the best.**

Vanilla RoBERTa (no fine-tuning).

→ **Fine-tuning (distillation) is necessary.**



Conclusion

- We propose **RelBERT**, a framework to achieve relation embedding model based on pretrained LM.
- RelBERT **distil the LM's relational knowledge** and realize a **high quality relation embedding**.
- Experimental results show that **RelBERT embedding outperform existing baselines**, establishing **SotA in analogy and relation classification.**

Release of RelBERT Library

We release python package **relbert** (install via `pip install relbert`) along with model checkpoints on the huggingface modelhub.

Please check our project page https://github.com/asahi417/relbert !!

```
from relbert import RelBERT
model = RelBERT('asahi417/relbert-roberta-large' )
# the vector has (1024,)
v_tokyo_japan = model.get_embedding([ 'Tokyo', 'Japan'])
```

🌳 Thank you! 🌳

Comparing to Word Embeddings

FastText is still better than RelBERT in Google Analogy Question.

Breakdown per relation types shows that FastText is better in the morphological relation, while very poor in the lexical relation.

Model	Google		BATS		
	Mor	**Sem**	**Mor**	**Sem**	**Lex**
FastText	95.4	98.1	90.4	71.1	33.8
RelBERT	89.8	95.8	87.0	66.2	75.1

Nearest Neighbours

Target	Nearest Neighbors
barista:coffee	baker:bread, brewer:beer, bartender:cocktail, winemaker:wine, bartender:drink, baker:cake
bag:plastic	bottle:plastic, bag:leather, container:plastic, box:plastic, jug:glass, bottle:glass
duck:duckling	chicken:chick, pig:piglet, cat:kitten, ox:calf, butterfly:larvae, bear:cub
cooked:raw	raw:cooked, regulated:unregulated, sober:drunk, loaded:unloaded, armed:unarmed, published:unpublished
chihuahua:dog	dachshund:dog, poodle:dog, terrier:dog, chinchilla:rodent, macaque:monkey, dalmatian:dog
dog:dogs	cat:cats, horse:horses, pig:pigs, rat:rats, wolf:wolves, monkey:monkeys
spy:espionage	pirate:piracy, robber:robbery, lobbyist:lobbying, scout:scouting, terrorist:terrorism, witch:witchcraft

Fine-tuning on Triples

Given a triple of the anchor x_a (eg. "`sunscreen`"), the positive x_p (eg. "`sunburn`"), and the negative x_n (eg. "`evil`"), the **triplet loss** is defined as

$$L_t = \max\left(0, \|x_a - x_p\| - \|x_a - x_n\| + \varepsilon\right)$$

and the **classification loss** is defined as

$$L_c = -\log(g(x_a, x_p)) - \log(1 - g(x_a, x_n))$$

$$g(u, v) = \mathrm{sigmoid}(W \cdot [u \oplus v \oplus |v - u|]^T)$$

where W is a learnable weight. The loss functions are inspired by SBERT (Reimers 2019).