Neural networks for NLP: Can structured knowledge help?

Christophe Gravier

Riken AIP

Tokyo

July 16th, 2019

Who am I

Acknowledgement

I am very thankful to the French Embassy for the travel grant I was offered !

- Associate Professor with habilitation
- Université Jean Monnet in Saint-Etienne
- Laboratoire Hubert Curien, UMR CNRS 5516
- Director of development and innovation at Télécom Saint-Etienne (U. Jean Monnet and affiliated to Institut Mines Télécom)

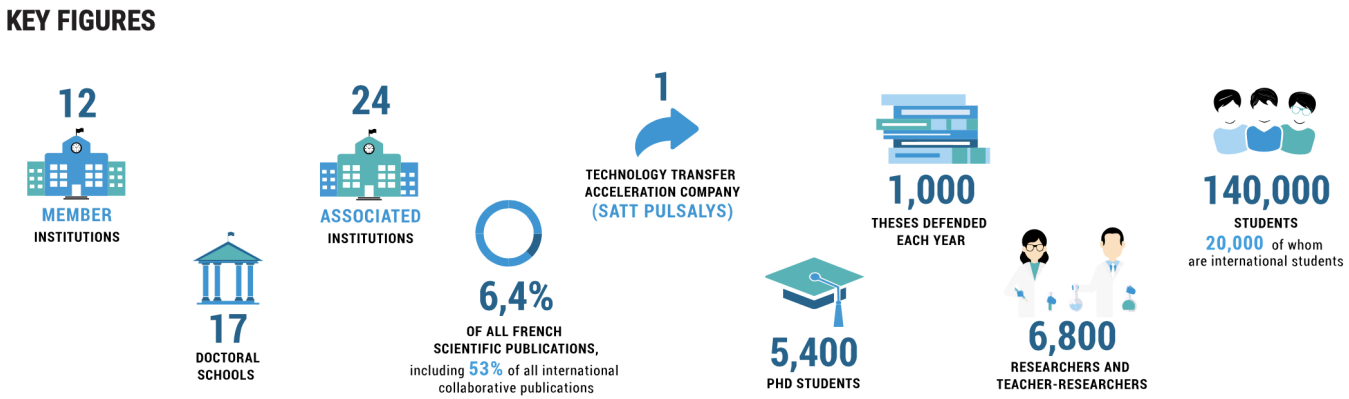
Research interests (NLP) :
Learning representations: improve embeddings, differents embeddings spaces such as Hamming or Wasserstein.
Downstream tasks: Question answering/generation, chatbots, free texts / structured taxonomies alignments, text generation.

Where I am coming from

Where I am coming from

Where I am coming from

Where I am coming from



People involved in this research

Julien Tissier

+ 3rd year PhD student
+ Currently at Riken AIP Tokyo
+ Co-supervised by Pr. **Amaury Habrard**, U. Jean Monnet, LaHC

Hady Elsahar

+ PhD student (defended on the 5th July)
+ Now at Naver Labs Grenoble
+ Co-supervised by Pr. **Frédérique Laforest**, INSA Lyon, Lab. LIRIS

European WDAqua Pierre&Marie Curie project, strong collaborations with U. of Southampton, especially **Lucie-Aimee Kaffee** and **Pavlos Vougiouklis** (and supervisors **Elena Simperl**, **Jonathon Hare**)

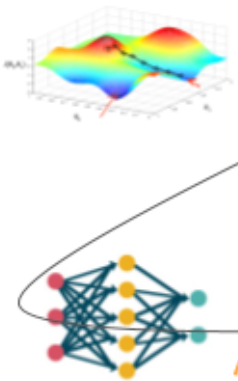
General modern NLP pipeline

1- Semi-supervised learning on very large textual corpus

Using datasets...
Wikipédia, open access books, ...

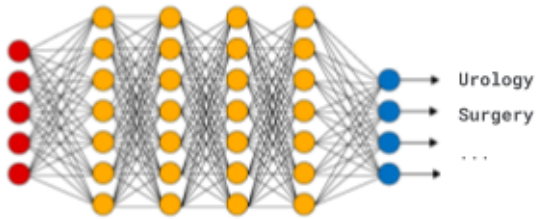
... We train a network to :
Predict a the next token in a sentence, a hidden word in a window, choosing the next next sentence between two given...

... We get :
Word representations (embeddings). The learnt model is discarded but we use the hidden layer holding weights for each words -- the latent word representation.



2- Supervised learning for a specific task (ex: text classification, intent detection and slot filing for chatbots, ...) using labeled data.

Description	Type
Whole body radionuclide bone scan due to prostate cancer.	Urology
Combined closed vitrectomy with membrane peeling, fluid-air exchange, and endolaser, right eye.	Surgery
Fertile male with completed family. Elective male sterilization via bilateral vasectomy.	Urology
...	...



Neural net 1 : Learn representations

Neural net 2 : Downstream task

Where are we know ? (GLUE [Wang et al. 2018])

GLUE SuperGLUE Tasks Leaderboard FAQ Diagnostics Submit

Rank	Name	Model	URL	Score	CoLA	SST-2	MRPC	STS-B	QQP	MNLI-m	MNLI-mm	QNLI	RTE	WNLI	AX
1	GLUE Human Baselines	GLUE Human Baselines		87.1	66.4	97.8	86.3/80.8	92.7/92.6	59.5/80.4	92.0	92.8	91.2	93.6	95.9	-
2	Microsoft D365 AI & MSR	MT-DNN-ensemble		84.2	65.4	96.5	92.2/89.5	89.6/89.0	73.7/89.9	87.9	87.4	96.0	85.7	65.1	42.8
3	王玮	ALICE large (Alibaba DAMO NLP		83.9	65.3	95.2	92.0/89.3	90.3/89.4	74.1/90.5	88.0	87.7	95.7	83.1	65.1	43.6
4	Stanford Hazy Research	Snorkel MeTaL		83.2	63.8	96.2	91.5/88.5	90.1/89.7	73.1/89.9	87.6	87.2	93.9	80.9	65.1	39.9
5	张倬胜	SemBERT		82.9	62.3	94.6	91.2/88.3	87.8/86.7	72.8/89.8	87.6	86.3	94.6	84.5	65.1	42.4
6	Anonymous Anonymous	BERT + BAM		82.3	61.5	95.2	91.3/88.3	88.6/87.9	72.5/89.7	86.6	85.8	93.1	80.4	65.1	40.7
7	Nitish Shirish Keskar	Span-Extractive BERT on STILTs		82.3	63.2	94.5	90.6/87.6	89.4/89.2	72.2/89.4	86.5	85.8	92.5	79.8	65.1	28.3
8	Jason Phang	BERT on STILTs		82.0	62.1	94.3	90.2/86.6	88.7/88.3	71.9/89.4	86.4	85.6	92.7	80.1	65.1	28.3
9	Jacob Devlin	BERT: 24-layers, 16-heads, 1024-		80.5	60.5	94.9	89.3/85.4	87.6/86.5	72.1/89.3	86.7	85.9	92.7	70.1	65.1	39.6
10	Neil Houlsby	BERT + Single-task Adapters		80.2	59.2	94.3	88.7/84.3	87.3/86.1	71.5/89.4	85.4	85.0	92.4	71.6	65.1	9.2
11	GLUE Baselines	BiLSTM+ELMo+Attn		70.0	33.6	90.4	84.4/78.0	74.2/72.3	63.1/84.3	74.1	74.5	79.8	58.9	65.1	21.7
		BiLSTM+ELMo		67.7	32.1	89.3	84.7/78.0	70.3/67.8	61.1/82.6	67.2	67.9	75.5	57.4	65.1	21.3
		Single Task BiLSTM+ELMo+Attn		66.5	35.0	90.2	80.2/68.8	55.5/52.5	66.1/86.5	76.9	76.7	76.7	50.3	65.1	27.9

BiLSTM+Attn: 65.6
...
InferSent: 63.9
...
CBOW: 58

All we need ...

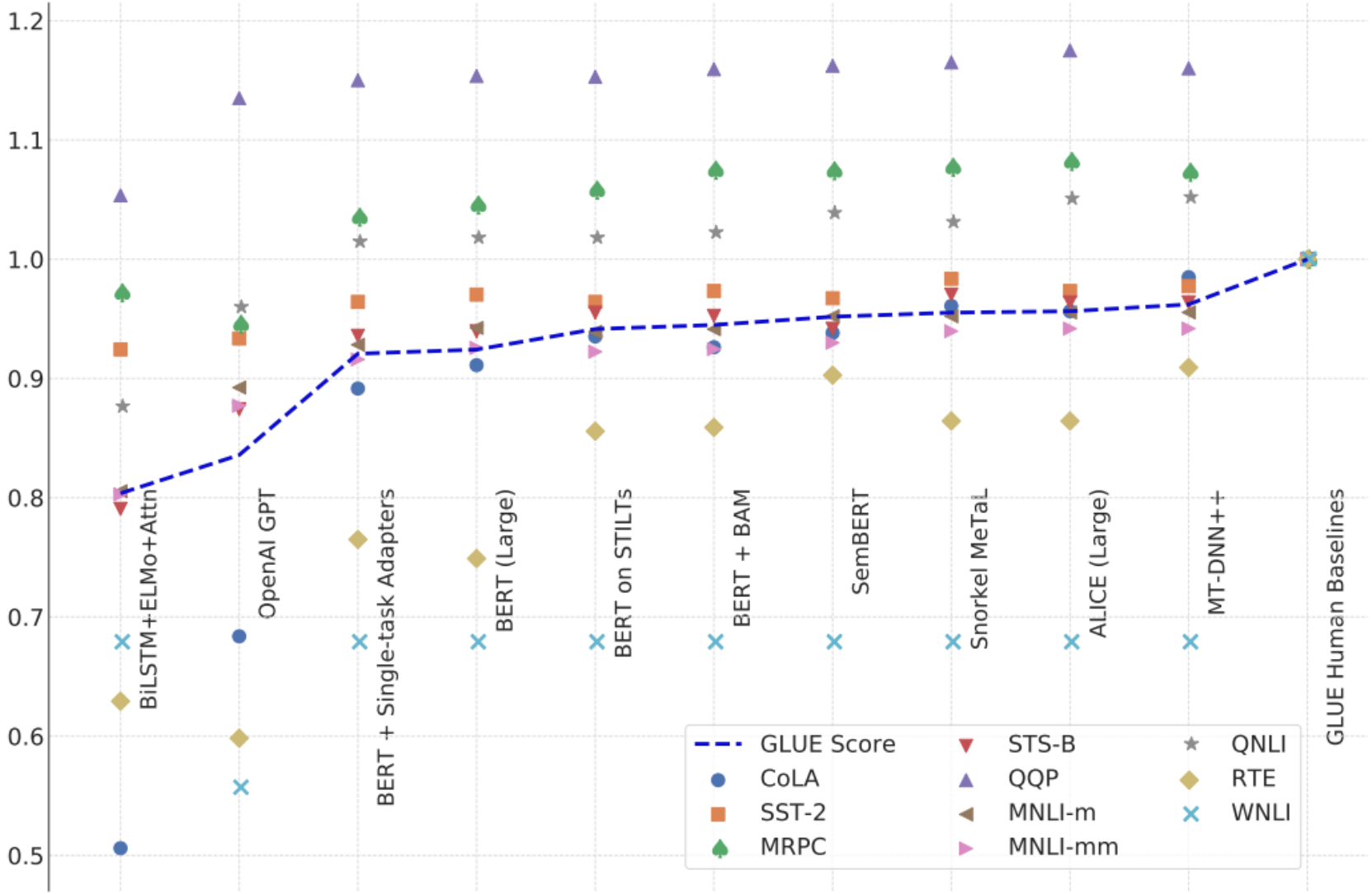
- Self-supervised learning from massive unlabelled text corpora
- Attention (and residual connections) is all you need (ELMO [Peters et al. 2018], BERT [Devlin et al. 2018])
- And Super GLUE [Wang et al. 2019])

All we need ...

- Self-supervised learning from massive unlabelled text corpora
- Attention (and residual connections) is all you need (ELMO [Peters et al. 2018], BERT [Devlin et al. 2018])
- And Super GLUE [Wang et al. 2019])

But. . . "Do Supervised Distributional Methods Really Learn Lexical Inference Relations" [Levy et al. 2015] ?

Wait ... (Figure from [Wang et al. 2019])



Coincidence ? I think not.

Most tasks present over-human performance ! But ...

- **No improvement** on Natural Language inference (WNLI) [Levesque, Davis, and Morgenstern 2012]
- **Subpar** for Recognizing Textual Entailment (RTE) [Dagan, Glickman, and Magnini 2005; Bar-Haim et al. 2006; Giampiccolo et al. 2007; Bentivogli et al. 2009]

Example from RTE (Does A entails hypothesis B ?):

- A: Time Warner is the worlds largest media and internet company
- B: Time Warner is the worlds largest company

Example from WNLI (Co-reference resolution):

- The trophy does not fit in the brown suitcase because **it** is too small. What is too small ? A: The trophy, B: The suitcase

Some unsolved problems (we are saved)

Some NLP models that are not "solved" [Devlin 2019]

- Models minimizing total training cost vs. accuracy on modern hardware
- **Models that are very parameter efficient (e.g., for mobile)**
- **Models that represent knowledge/context in latent space**
- Models that represent structured data (e.g., knowledge graph)
- Models that jointly represent vision and language

What we are gonna discuss today

How to combine the modern neural based NLP pipeline with external, structured (or semi-structured) knowledge and how much does it help ?

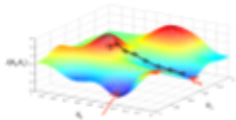
Table of contents

Plain old dictionaries

1- Semi-supervised learning on very large textual corpus

Using datasets...
Wikipédia, open access books, ...

... We train a network to :
Predict the next token in a sentence, a hidden word in a window, choosing the next sentence between two given...



... We get :
Word representations (embeddings).
The learnt model is discarded but we use the hidden layer holding weights for each words -- the latent word representation.

Neural net 1 :
Learn representations

+

Word2vec

- Slide a window across a corpora
- Move closer vectors of words within the same window

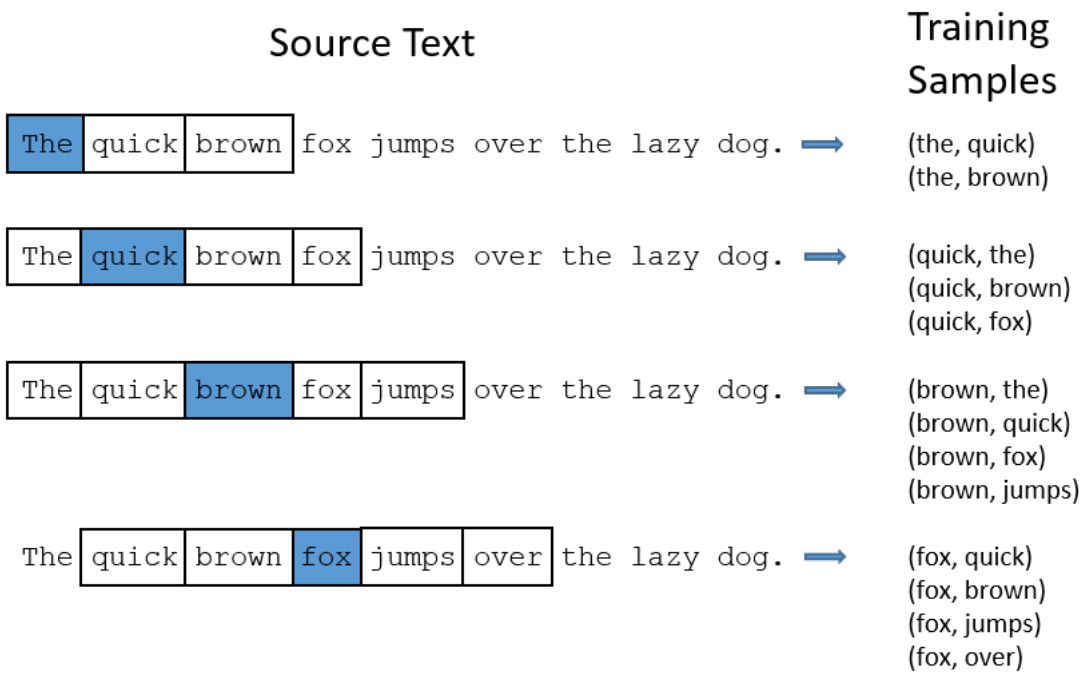


Image taken from : `https://mubaris.com/posts/word2vec/`

word2vec

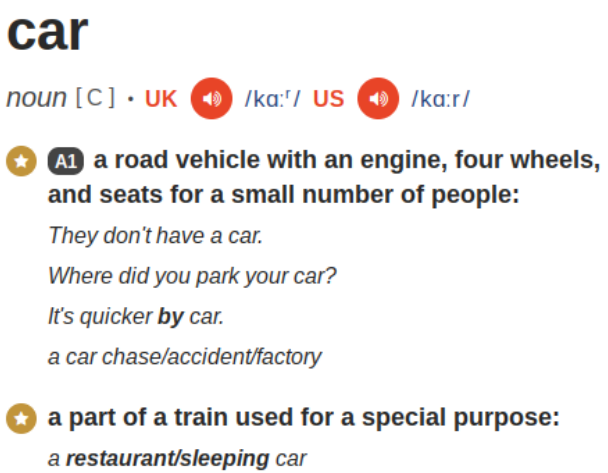
Limitations

- Words within the same window **are not always related**
- Synonymy, meronymy, etc. **happen rarely** inside a window

Solution: add information contained in dictionaries

+ **Strong semantic** information
+ **Weak supervision** using higher-level and/or noisier input from subject matter experts [Ratner et al. 2017], here linguists.
+ Move closer related words

Crawling online dictionaries



car

noun [C] · UK /kɑːʳ/ US /kɑːr/

A1 **a road vehicle with an engine, four wheels, and seats for a small number of people:**

They don't have a car.

Where did you park your car?

*It's quicker **by** car.*

a car chase/accident/factory

a part of a train used for a special purpose:

*a **restaurant/sleeping** car*

- Crawl dictionary webpage of each word
- Parse HTML to extract the definitions
- Use 4 different dictionaries (Oxford, Cambridge, Collins, dictionary.com) $\rightarrow$ Definitions of 200k words
- For each word, all senses are concatenated (no disambiguation)
- POS, etymology, synonymy ... are not taken into account

Strong and Weak pairs

car : a road vehicle, typically with four wheels, powered by an internal combustion engine and able to carry a small number of people.
vehicle : a thing used for transporting people or goods, especially on land, such as a car, lorry or cart.
wheel : a circular object that revolves on an axle and is fixed below a vehicle or other object to enable it to move easily over the ground.

- **Strong pair**: mutual inclusion (car – vehicle)
- **Weak pair**: one-sided inclusion (car – wheel), (wheel – vehicle)

Objective function

Positive Sampling: move closer words forming either a strong or a weak pair

$$J_{pos}(w_t) = \beta_s \sum_{w_i \in S_{trong}(w_t)} \ell(v_t \cdot v_i) + \beta_w \sum_{w_j \in W_{eak}(w_t)} \ell(v_t \cdot v_j) \quad (1)$$

Controlled Negative Sampling: prevent related words to be moved further

$$J_{neg}(w_t) = \sum_{\substack{w_i \in \mathcal{F}(w_t) \\ w_i \notin S_{trong}(w_t) \\ w_i \notin W_{eak}(w_t)}} \ell(-v_t \cdot v_i) \quad (2)$$

Global Objective Function

$$J = \sum_{t=1}^{C} \sum_{c=-n}^{n} \ell(v_t \cdot v_{t+c}) + J_{pos}(w_t) + J_{neg}(w_t) \quad (3)$$

Word semantic similarity evaluation

word1	word2	similarity (mean)
tiger	cat	7.35
book	paper	7.46
computer	keyboard	7.62
plane	car	5.77
telephone	communication	7.50
stock	phone	1.62
stock	egg	1.81
fertility	egg	6.69
Arafat	Jackson	2.50
law	lawyer	8.38

Table: Example of WS353 word similarity dataset.

Word semantic similarity evaluation

- Trained on Wikipedia (July 2017), 3 corpora sizes
- Compared to word2vec [Mikolov et al. 2013] (w2v) and fasttext [Bojanowski et al. 2017] (FT)

	50M			200M			Full		
	w2v	FT	our	w2v	FT	our	w2v	FT	our
MC-30	69.7	72.2	**84.0**	74.2	79.5	**85.4**	80.9	83.1	**86.0**
RW	37.5	44.2	**47.5**	37.7	**47.5**	46.7	40.7	46.4	**48.2**
SimVerb	16.5	17.9	**36.3**	18.3	20.6	**37.7**	18.6	22.2	**38.4**
WS353	66.0	65.7	**73.8**	69.4	70.1	**76.2**	70.5	72.9	**75.6**
YP-130	45.8	41.5	**66.6**	44.9	50.9	**61.6**	50.2	53.3	**64.6**

Table: Spearman's rank correlation scores for several similarity datasets.

Using WordNet pairs for positive sampling

- Replace strong pairs with WordNet pairs
- Dictionaries are better than WordNet for positive sampling

	50M	200M	full
No pairs	45.3	47.1	48.8
With WordNet pairs	56.4	56.6	55.9
With dictionary pairs	**56.9**	**56.9**	**57.1**

Table: Weighted average semantic similarity score of dict2vec vectors when trained with WordNet or dictionary pairs.

Using dictionaries for retrofitting

- Retrofitting: moving closer related words **after** training as in [Faruqui et al. 2014]
- Dictionaries are also better than WordNet for retrofitting
- Retrofitted w2v and FT are worse than the basic dict2vec vectors

	50M			200M			Full		
	w2v	FT	our	w2v	FT	our	w2v	FT	our
No retrofit	45.3	46.7	56.9	47.1	50.3	56.9	48.8	50.8	57.1
$R_{WordNet}$	47.4	47.6	58.2	48.8	50.4	58.1	50.7	50.3	58.3
$R_{dictionary}$	**47.9**	**48.9**	**58.2**	**49.4**	**52.4**	**58.7**	**51.2**	**52.9**	**59.2**

Table: Weighted average semantic similarity score of raw vectors and after retrofitting with WordNet or dictionary pairs.

Conference reviewer

"Comparison is not fair, word2vec and Fasttext never have a chance to see the dictionary data. You should at least try to add the dictionary content to the wikpiedia dump ?" (Reviewer2)

Recap

- Dictionaries contain **important semantic information**
- **Positive sampling** allows to incorporate additional information during learning (dictionaries or WordNet)
- If **retrofitting** is desirable in your application, use dictionary over Wordnet.
- **Clamping domain corpora** to a Wikipedia dump for training may be a very cheap boost for domain-specific downstream tasks.
- You don't have to think big to make something count.

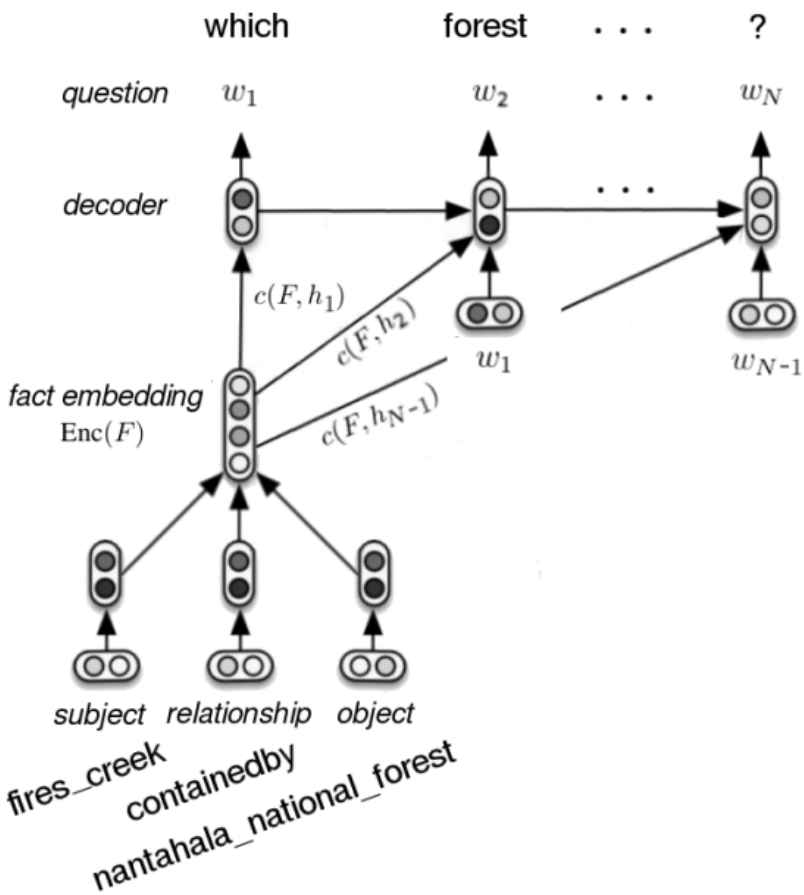
Entire source code for training and downloading pretrained vectors:
`https://github.com/tca19/dict2vec`

Table of contents

Question Generation from Knowledge Graphs

From [Serban et al. 2016]:

- Given a Knowledge base triple [Subject, Predicate, Object]
- Generate a question that asking about the object of the triple
- Feed-forward sequence-to-sequence decoder architecture



Training

Trained using datasets aligning questions (free text) and triples.
Most popular dataset: SimpleQuestions dataset [Bordes et al. 2015] (Freebase KB).

Training

Trained using datasets aligning questions (free text) and triples.
Most popular dataset: SimpleQuestions dataset [Bordes et al. 2015] (Freebase KB).

Examples:

(fb:fires_creek, fb:containedby, fb:nantahala_national_forest)

Which forest is `fires_creek` in?

(fb:jimmy_neutron, fb:fictional_character_occupation, fb:inventor)

What does `jimmy_neutron` do?

(fb:kimchi, fb:incompatible_with_dietary_restrictions, fb:veganism)

What dietary restriction is incompatible with `kimchi` ?

Wait...

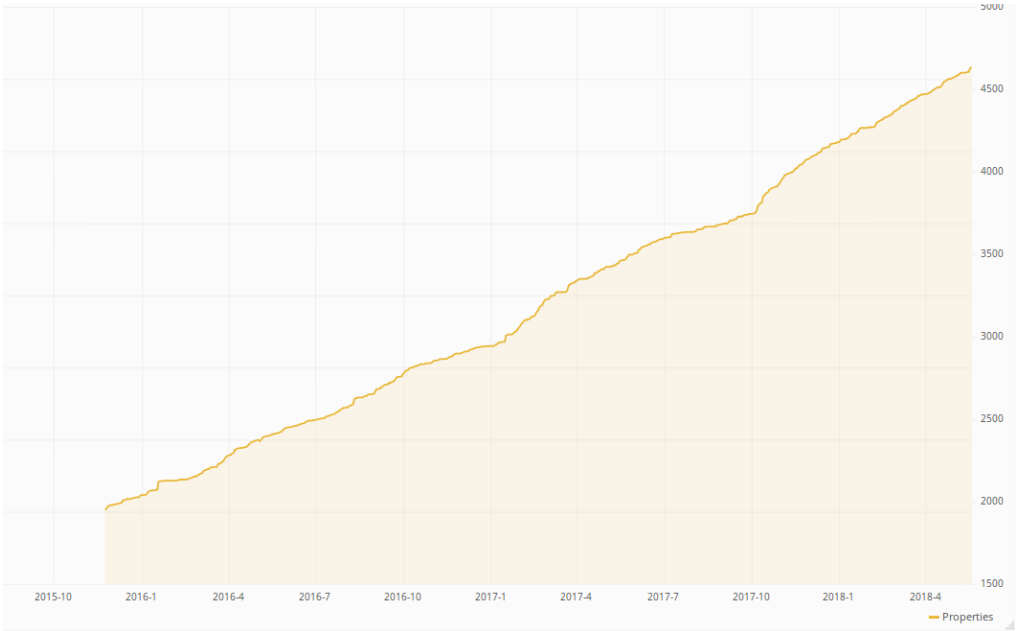
We usually want to answer questions not the contrary!

Question Generation has been shown to improve the performance of factoid Question Answering Systems either by **dual training** or by **augmenting existing training datasets** [Dong et al. 2017; Reddy et al. 2017]

Problems

Problem 1: Coverage and bias.
75% of Freebase predicates are **not** in SimpleQuestions.

Problem 2: Predicates sets size have usually a monotonic increase.
Extreme case: 2500+ Wikidata predicates over the last 3 years.



How to generalize of QA systems beyond training datasets ?

What happens if we do nothing

Generate the question given the input :

Unseen predicate: `[videogame publisher]`
unseen entities (`[Tokyo Xtreme Racer]`, `[SEGA]`).

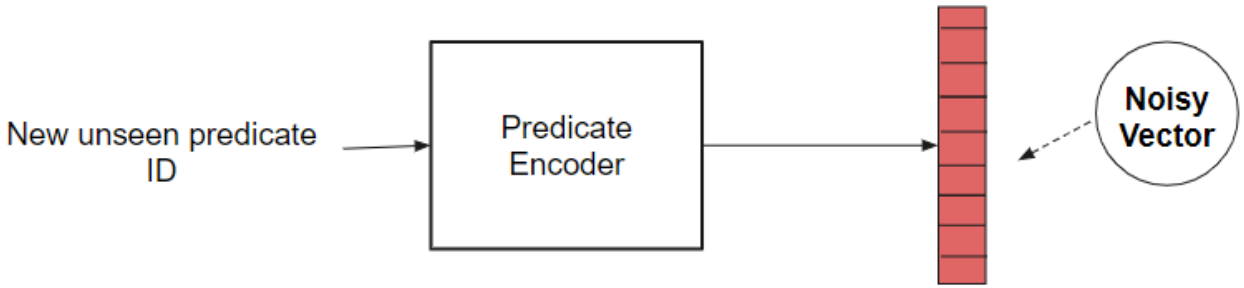
Note: unseen predicate means we do not have `[videogame publisher]` in our predicate vocabulary – we did not even saw any instances of that class in the training set.

Expected generated questions

- what is the name of the topic of the film tokyo xtreme racer?
- what is the videogame of the computer videogame tokyo xtreme racer ?
- what is the developer of the computer videogame tokyo xtreme racer ?
- what is the genre of the album tokyo xtreme racer ?
- what is the name of the tokyo xtreme racer ?
- which country is known for playing by tokyo xtreme racer ?
- what kinds of videogame is tokyo xtreme racer ?

Explanations

Explanation #1 : The encoder fails to encode the new unseen IDs into meaningful vectors. The decoder is not able to pair those vectors to words.



$\Rightarrow$ Garbage in, garbage out: it generates rubbish.

Explanations

Explanation #2 : The rare word problem.
The following words are rare in the training set so they have been generated with low probability in the decoder language model – it is unlikely that they will be used to generate questions :

- videogame
- company
- released
- game

So instead the decoder generates:

What is the genre of the `album` tokyo xtreme racer ?

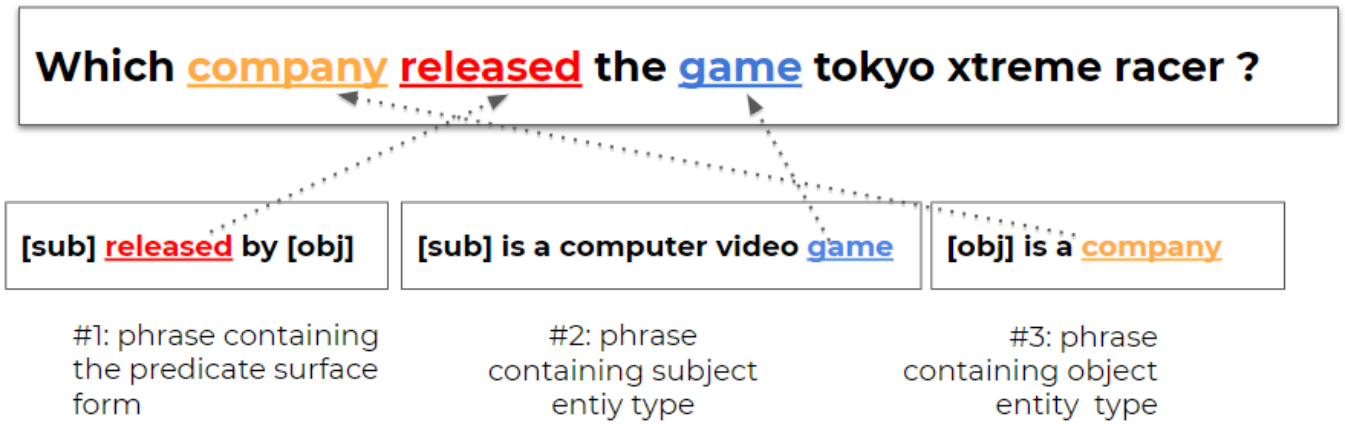
Note : Remember the main semantic content in Freebase. . .

Textuals contexts

Intuitively, a human who is given the task to write a question on a fact offered by a KB, would read natural language sentences where the entity or the predicate of the fact occur, and build up questions that are aligned with what he reads from both a lexical and grammatical standpoint.

Intuition

Providing models with **textual contexts** that have potentially high overlap with the target Question – they can provide semantic and vocabulary richness to the decoder.



New inputs

We start with a new set of inputs for the encoder (triple + textual contexts) :

Input Triple

fb:/m/07_3zt [Tokyo Xtreme Racer]
fb:/cvg/computer_videogame/publisher
fb:/m/09ymx4 [SEGA]

Textual contexts:

[sub] released by [obj]

[obj] is a company

[sub] is a computer video game

Collecting textual contexts

T-REx : A Large Scale Alignment of Natural Language with Knowledge Base Triples [LREC2018]

`hadyelsahar.github.io/t-rex`

Dataset	Documents / Format	Unique predicates	Aligned Triples	Availability
NYT-FB (yao et al. 2011)	1.8M Sent.	258	39K	partially available
TAC KBP	90K Sent.	41	122K	closed
FB15K-237	2.7 M textual-relations	237	2.7M	publicly available
Wikireadings	4.7M Articles	884	n.a.	publicly available
Google-RE	60K Sent.	5	60K	publicly available
T-REx	**6.2M Sent.**	**642**	**11M**	**publicly available**

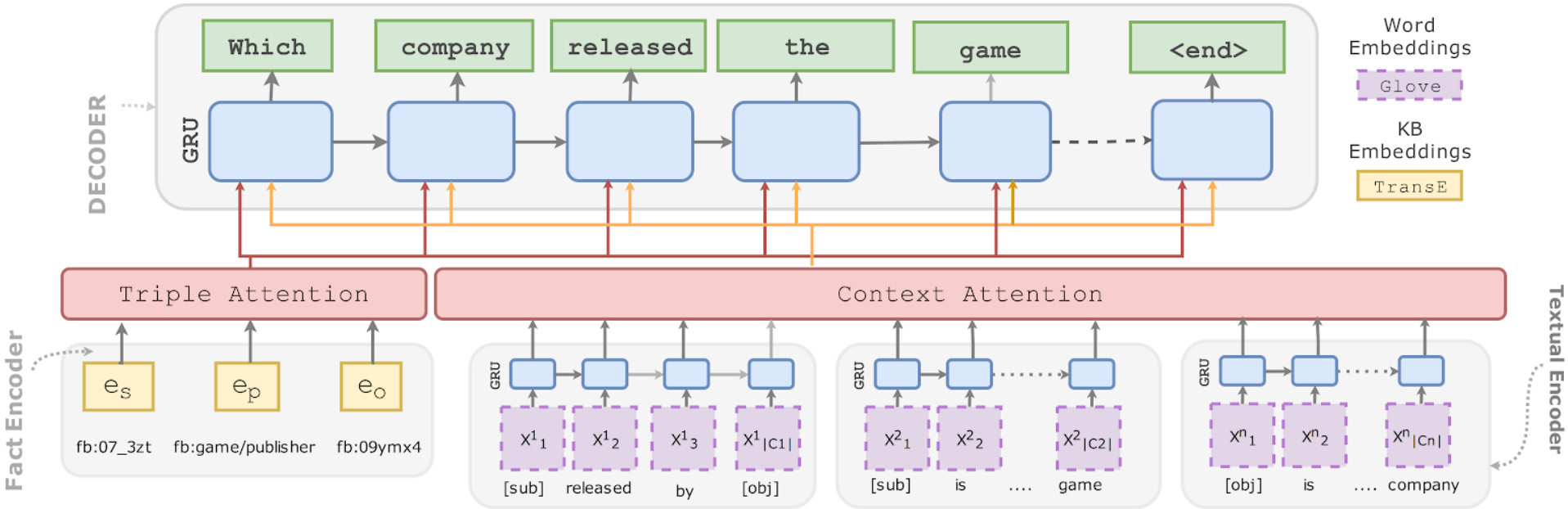
Textual contexts from T-Rex

Examples :

Freebase Relation	Predicate Textual Context
person/place_of_birth	[O] is birthplace of [S]
currency/former_countries	[S] was currency of [O]
dish/cuisine	[O] dish [S]
airliner_accident/flight_origin	[S] was flight from [O]
film_featured_song/performer	[S] is release by [O]
airline_accident/operator	[S] was accident for [O]
genre/artists	[S] became a genre of [O]
risk_factor/diseases	[S] increases likelihood of [O]
book/illustrations_by	[S] illustrated by [O]
religious_text/religion	[S] contains principles of [O]
spacecraft/manufacturer	[S] spacecraft developed by [O]

Find them: plain string matching and then replace subject by [S] and object by [O] to keep track of the direction.

Zero-shot QG from triples: neural architecture



– Facts embeddings are initialized using Trans-E
– Textual encoders words embeddings using Glove.

Dealing with out-of-vocabulary and rare words

Copy action mechanism [Luong et al. 2014] using POS.

– Each word in every input textual context is replaced by a special token containing a combination of its context id (e.g. C1) and its POS tag (e.g. NOUN).
– Then, update the question with the corresponding tag if possible

What caused the [C1_NOUN] of the [C3_NOUN] [S] ?	
C1	[S] **death** by [O]
	[S] **[C1_NOUN]** [C1_ADP] [O]
C2	Disease
	[C2_NOUN]
C3	Musical **artist**
	[C3_ADJ] **[C3_NOUN]**

Inference regime

At each time step the network draw a token from the vocabulary or a placeholder – the latter being replaced with their original words from the textual contexts as a final step.

Baselines

SELECT: baseline from [Serban et al. 2016].
Unseen predicate ? Pick the question from the training set that corresponds to the triple having the same answer type.
Unseen sub-types or obj-types ? Pick the fact that have the same answer predicate.

R-TRANSE: Same as above but we select the fact in the training set with the closest cosine similarity (concatenation of TransE embeddings).

IR: A historically IR-based solution using TD-IDF and LSA to pick up questions to ouput from the training set.

Encoder-Decoder: Encoder-Decoder model with a single placeholder (best model in [Serban et al. 2016]). Same embeddings initialization as our.

Results for unseen predicates

Standard metrics from the machine translation task.
In this case we measure how close each generated question is close to its "ground truth" question.

	Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE_L	METEOR
Unseen Predicates	SELECT	46.81 ± 2.12	38.62 ± 1.78	31.26 ± 1.9	23.66 ± 2.22	52.04 ± 1.43	27.11 ± 0.74
	IR	48.43 ± 1.64	39.13 ± 1.34	31.4 ± 1.66	23.59 ± 2.36	52.88 ± 1.24	27.34 ± 0.55
	IR_{COPY}	48.22 ± 1.84	38.82 ± 1.5	31.01 ± 1.72	23.12 ± 2.24	52.72 ± 1.26	27.24 ± 0.57
	R-TRANSE	49.09 ± 1.69	40.75 ± 1.42	33.4 ± 1.7	25.97 ± 2.22	54.07 ± 1.31	28.13 ± 0.54
	$\text{R-TRANSE}_{\text{COPY}}$	49.0 ± 1.76	40.63 ± 1.48	33.28 ± 1.74	25.87 ± 2.23	54.09 ± 1.35	28.12 ± 0.57
	Encoder-Decoder	58.92 ± 2.05	47.7 ± 1.62	38.18 ± 1.86	28.71 ± 2.35	59.12 ± 1.16	34.28 ± 0.54
	Our-Model	60.8 ± 1.52	49.8 ± 1.37	40.32 ± 1.92	30.76 ± 2.7	60.07 ± 0.9	35.34 ± 0.43
	$\text{Our-Model}_{\text{copy}}$	**62.44** ± 1.85	**50.62** ± 1.46	**40.82** ± 1.77	**31.1** ± 2.46	**61.23** ± 1.2	**36.24** ± 0.65

Evaluation results of our model and all other baselines for the **unseen predicate** evaluation setup.

Results for unseen entity types

	Model	BLEU-4	ROUGE_L
Sub-Types	R-TRANSE	32.41 ± 1.74	59.27 ± 0.92
	Encoder-Decoder	42.14 ± 2.05	68.95 ± 0.86
	Our-Model	42.13 ± 1.88	69.35 ± 0.9
	$\text{Our-Model}_{\text{copy}}$	**42.2** ± 2.0	**69.37** ± 1.0
Obj-Types	R-TRANSE	30.59 ± 1.3	57.37 ± 1.17
	Encoder-Decoder	37.79 ± 2.65	65.69 ± 2.25
	Our-Model	37.78 ± 2.02	65.51 ± 1.56
	$\text{Our-Model}_{\text{copy}}$	**38.02** ± 1.9	**66.24** ± 1.38

Automatic evaluation of our model against selected baselines for **unseen sub-types and obj-types**.

I'm BLEU – what ?

BLEU suffers from many limitations [Novikova et al. 2017] – as all automatic evaluation metrics.

I'm BLEU – what ?

BLEU suffers from many limitations [Novikova et al. 2017] – as all automatic evaluation metrics.

Given this question : "**What kind of film is kill bill vol. 2?**", which is the most similar question :

A (Bleu: 71) – What is the name of the film kill bill vol. 2 ?
B (Bleu: 55) – Which genre is kill bill vol. 2 in?

I'm BLEU – what ?

BLEU suffers from many limitations [Novikova et al. 2017] – as all automatic evaluation metrics.

Given this question : "**What kind of film is kill bill vol. 2?**", which is the most similar question :

A (Bleu: 71) – What is the name of the film kill bill vol. 2 ?
B (Bleu: 55) – Which genre is kill bill vol. 2 in?

Human evaluation :
– Four annotators
– 100 facts randomly chosen
– Evaluation for each 4 models

Human evaluation

Predicate identification: Generated question contains the given predicate in the fact (yes/no).

Naturalness: Following [Ngonga Ngomo et al. 2013] we measure the comprehensibility and readability of the generated questions on a 1–5 Likert scale.

Model	% Pred. Identified	Natural.
Encoder-Decoder	6	3.14
Our-Model (No Copy)	6	2.72
Our-Model$_{copy}$ (Types context)	**37**	**3.21**
Our-Model$_{copy}$ (All contexts)	**46**	2.61

Results of human evaluation on % of predicates identified and naturalness 0-5.

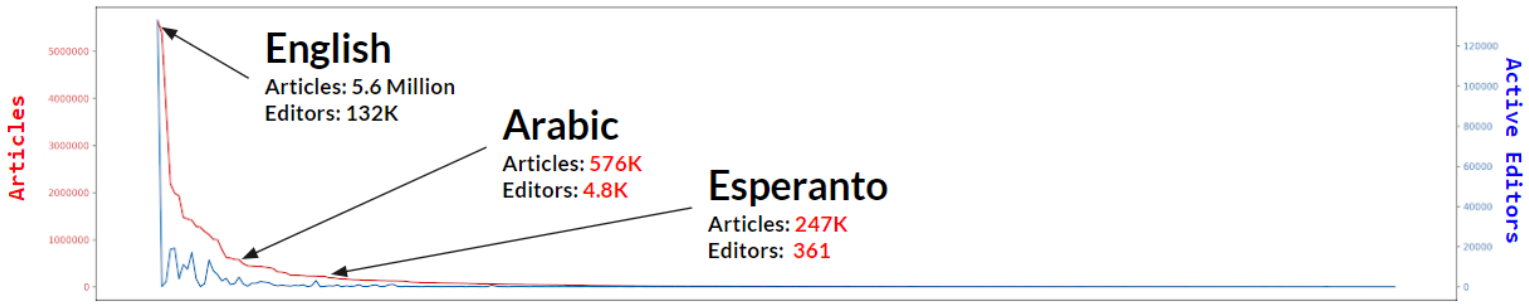
Table of contents

Wikipedia placeholder

When an entity lacks its Wikipedia page, **Wikidata is used to generate an article placeholder as content** (think underserved languages).

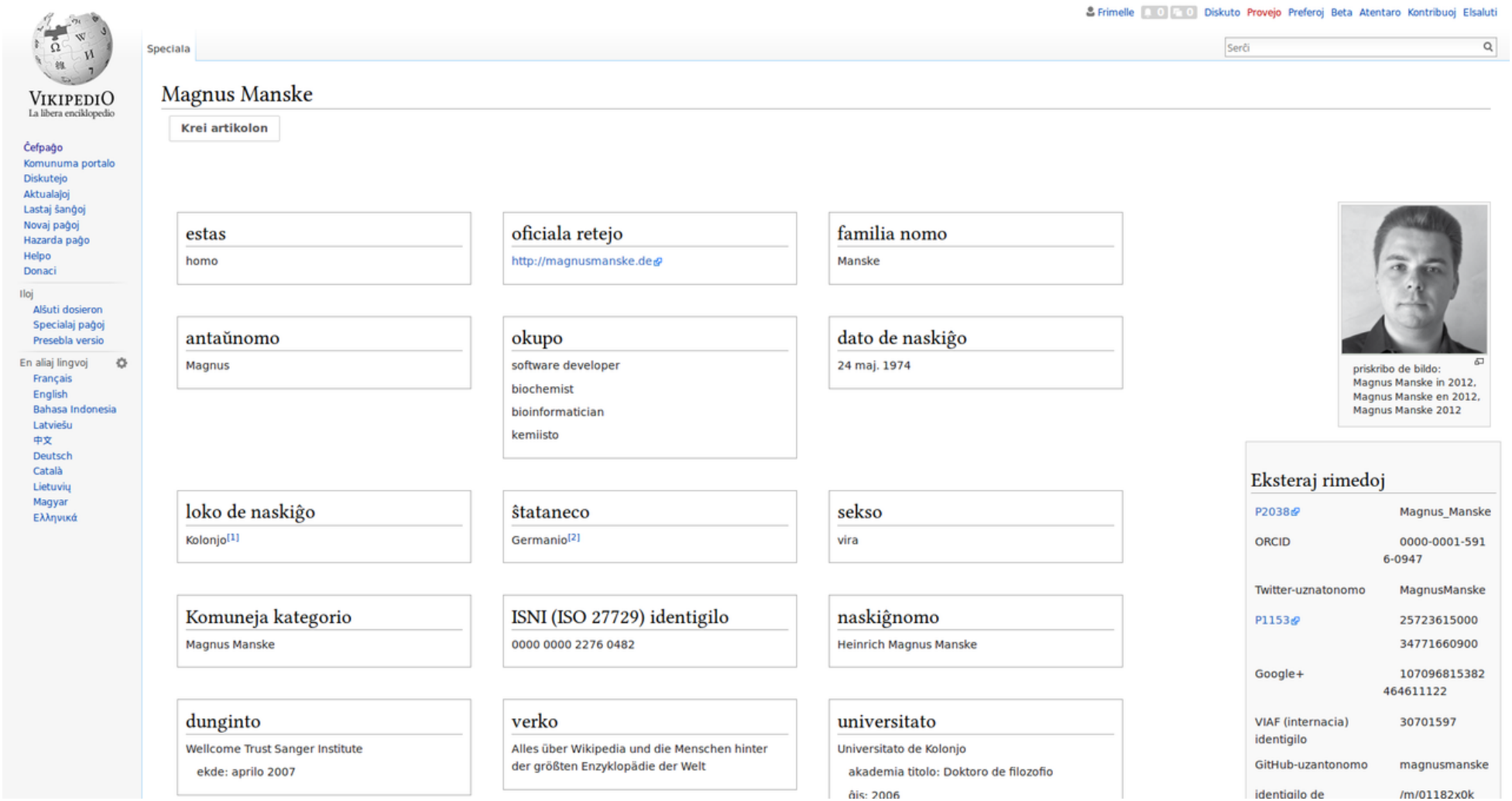
Display Wikidata triples in Wikipedia in a **tabular way** – with **multilingual** interlinked data from Wikidata.

Currently deployed on **14 under-resourced Wikipedias** (e.g. Haitian Creole, Urdu, Gujarati)



Access to knowledge !
Can we go further ?

Wikipedia placeholder



From ArticlePlaceholder to features

Given the following text :
"La Nigragora pigogarolo (Calocitta colliei) estas rimarkinda longvosta pigogarolo de la familio ()" (extract from Esperanto Wikipedia `https://eo.wikipedia.org/wiki/Nigragor%C4%9Da_pigogarolo`)

From ArticlePlaceholder to features

Given the following text :
"La Nigragora pigogarolo (Calocitta colliei) estas rimarkinda longvosta pigogarolo de la familio ()" (extract from Esperanto Wikipedia `https://eo.wikipedia.org/wiki/Nigragor%C4%9Da_pigogarolo`)

We leverage existing Wikidata triples :
`[S] [taksonomia nomo] [Calocitta]`
`[S] [supera taksono] [Pigogarolo]`
...

From ArticlePlaceholder to features

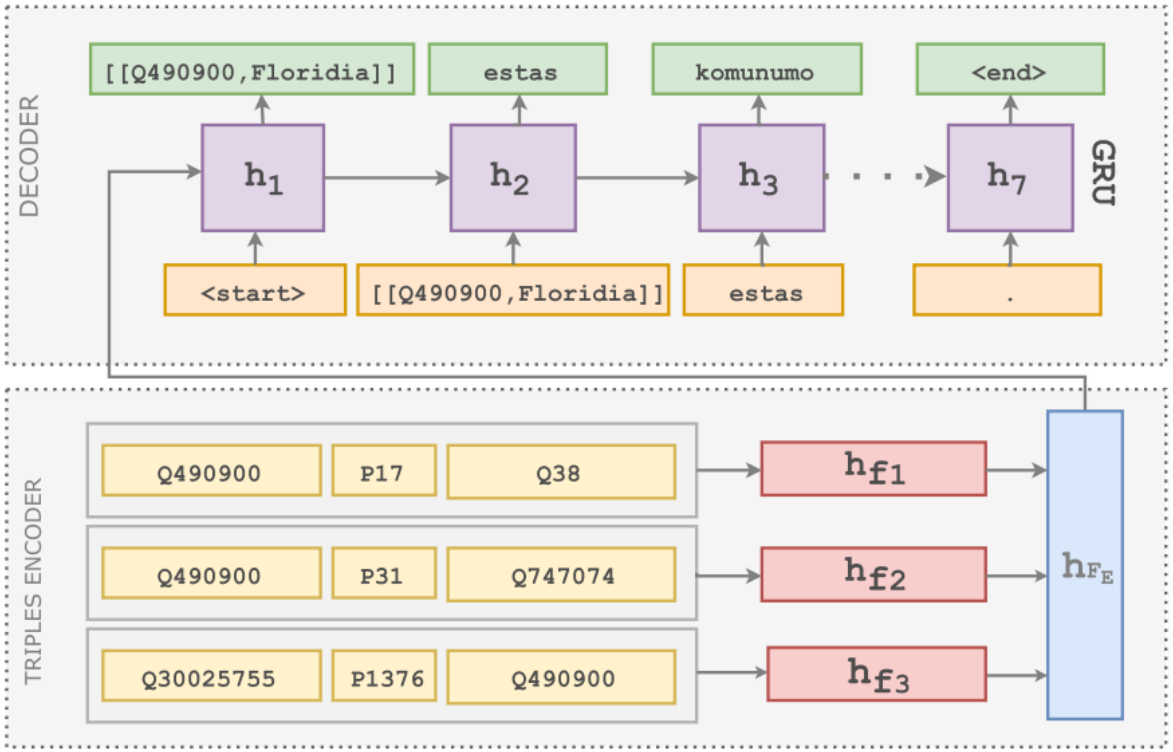
Given the following text :
"La Nigragora pigogarolo (Calocitta colliei) estas rimarkinda longvosta pigogarolo de la familio ()" (extract from Esperanto Wikipedia `https://eo.wikipedia.org/wiki/Nigragor%C4%9Da_pigogarolo`)

We leverage existing Wikidata triples :
`[S] [taksonomia nomo] [Calocitta]`
`[S] [supera taksono] [Pigogarolo]`
...

Then we replace the entity or predicate in the original sentence, but we use a property placeholder mechanism to tackle rare and oov words :
La Nigragora pigogarolo (`[[P225]]` colliei) estas rimarkinda longvosta `[[P171]]` de la familio...

Neural Wikipedian architecture

Same kind of mechanisms involved (without POS copy action and textual contexts that we discovered later) than in the previous Question Generation task.



Neural Wikipedian results

	Model	BLEU 1		BLEU 2		BLEU 3		BLEU 4		$ROUGE_L$		METEOR	
		Valid.	Test	Valid.	Test	Valid.	Test	Valid.	Test	Valid.	Test	Valid.	Test
Arabic	KN	12.84	12.85	2.28	2.4	0.95	1.04	0.54	0.61	17.08	17.09	29.04	29.02
	KN_{ext}	28.93	28.84	21.21	21.16	16.78	16.76	13.42	13.42	28.57	28.52	30.47	30.43
	IR	41.39	41.73	34.18	34.58	29.36	29.72	25.68	25.98	43.26	43.58	32.99	33.33
	IR_{ext}	49.87	48.96	42.44	41.5	37.29	36.41	33.27	32.51	51.66	50.57	34.39	34.25
	Ours	53.61	54.26	47.38	48.05	42.65	43.32	38.52	39.20	64.27	64.64	45.89	45.99
	+ Copy	**54.10**	**54.40**	**47.96**	**48.27**	**43.27**	**43.60**	**39.17**	**39.51**	**64.60**	**64.69**	**46.09**	**46.17**
Esperanto	KN	18.12	17.8	6.91	6.64	4.18	4.0	2.9	2.79	37.48	36.9	31.05	30.74
	KN_{ext}	25.17	24.93	16.44	16.3	11.99	11.92	8.77	8.79	44.93	44.77	33.77	33.71
	IR	43.01	42.61	33.67	33.46	28.16	28.07	24.35	24.3	46.75	45.92	20.71	20.46
	IR_{ext}	**52.75**	**51.66**	43.57	42.53	37.53	36.54	33.35	32.41	58.15	57.62	31.21	31.04
	Ours	49.34	49.40	42.83	42.95	38.28	38.45	34.66	34.85	66.43	**67.02**	40.62	**41.13**
	+ Copy	50.22	49.81	**43.57**	**43.19**	**38.93**	**38.62**	**35.27**	**34.95**	**66.73**	66.61	**40.80**	40.74

Standard automatic evaluation metrics results on test set of Wikipedia article summaries.

Human evalaution

Readers

– 27 participants (Wikipedia users) for each of the 2 studied languages (**Esperanto** and **Arabic**)
– **Fluency**: 0–6 score for text understandabilty and grammatically correct
– **Appropriateness** : Does the summary "feel" like a Wikipedia article ?

Experimental sentences set made of our generated sentences + sentences from Wikipedia itself and news feed.

Editors

Editors were asked to edit the article **starting from our summary** and the corresponding triples (2-3 sentences).

How much of the text was reused ?
The more, the best the summary. Measured using Greedy String-Tiling (GST) – detects whole bock moves unlike Levenstein distance.

Readers evaluation

		Fluency		Appropriateness
		Mean	SD	Part of Wikipedia
Arabic	Our model	4.7	1.2	77%0.77
	Wikipedia	4.6	0.9	74%0.74
	News	5.3	0.4	35%0.35
Esperanto	Our model	4.5	1.5	69%0.69
	Wikipedia	4.9	1.2	84%0.84
	News	4.2	1.2	52%0.52

Results for fluency and appropriateness as assessed by Wikipedian in the two underserved languages.

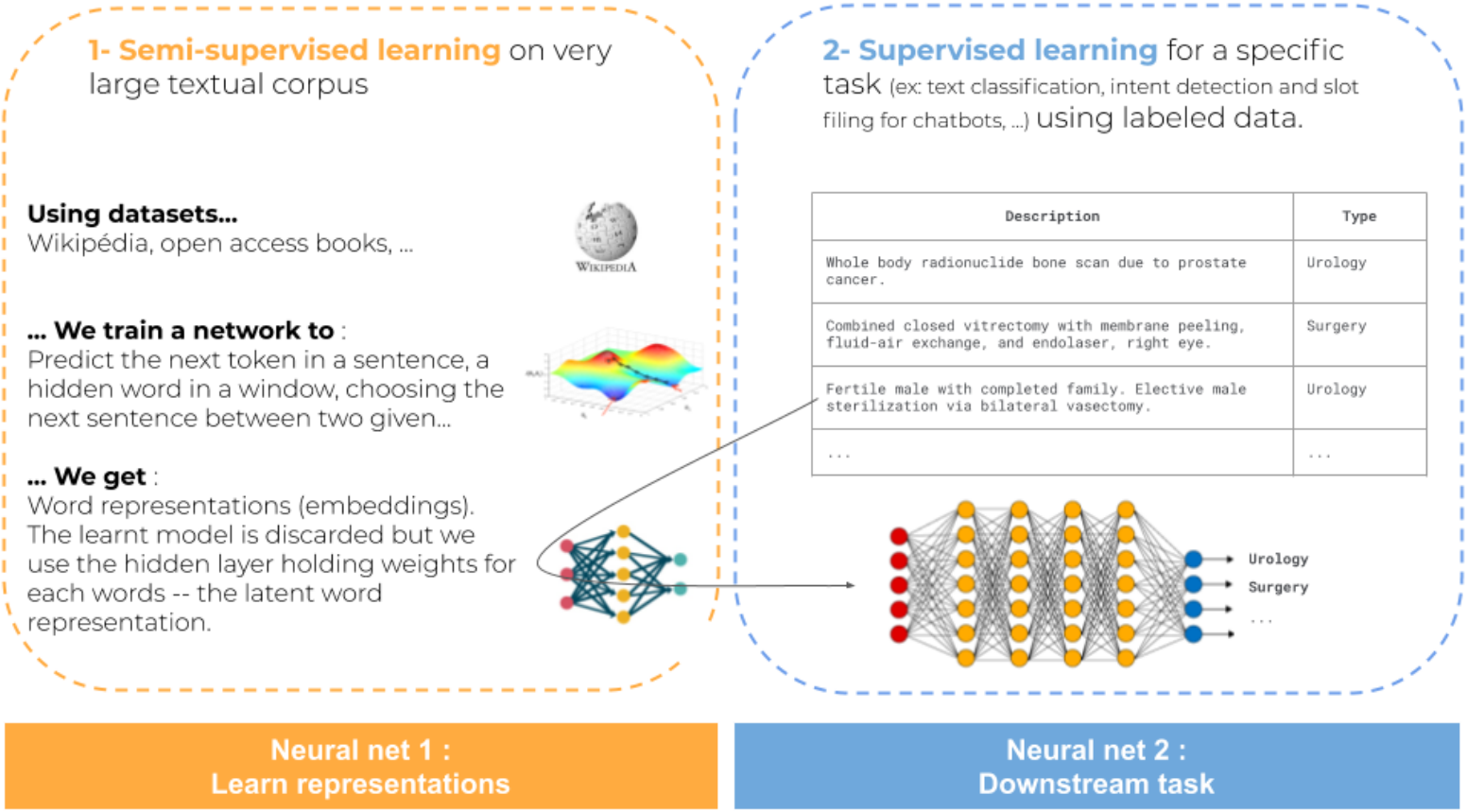
Editors evaluation

	Category	Examples	Percentage
Arabic	WD	.خماسي كلوريد الزرنيخ مركب كيميائي له الصيغة(كلمة ناقصة)، ويكون على شكل بلورات بيضاء خماسي كلوريد الزرنيخ هو مُركب كيميائي له الصيغة(AtClu2085)، ويكون على شكل بلورات بيضاء.	45.45%
	PD	بيتش باتوم أوهايو (بالإنجليزية (كلمة ناقصة) Ohio)هي منطقة سكنية تقع في الولايات المتحدة في(كلمة ناقصة.) بيتش باتوم (بالإنجليزية:Beach Batom)هي قرية تقع في الولايات المتحدة الامريكية في بروك كاونتي.	33.33%
	ND	دير علا هي بلدة تقع في جنوب غرب إيران. دير علا، أو بيثر، هي قرية أردنية	21.21%
Esperanto	WD	Zederik estas komunumo en la nederlanda provinco Zuid-Holland. Zederik estas komunumo en la nederlanda provinco Zuid-Hooland kaj estas ĉirkaŭata de la municipoj Lopik kaj Zederik.	78.98%
	PD	Nova Pádua estas municipo en la brazila subŝtato Suda Rio-Grando, kiu havis (manka nombro) loĝantojn en (jaro). Nova Pádua estas municipo en la brazila subŝtato Suda Rio-Grando.	15.79%
	ND	Ibiúna estas municipo de la brazila subŝtato San-Paŭlio, kiu taksis (manka nombro) enloĝantojn en (jaro). Ibiúna estas brazila [[municipo]] kiu troviĝas en la administra unuo [[San-Paŭlo]].	5.26%

Over 60 sampled evaluated summaries, 78% Whole or Partially reused for Arabic, 94% for Esperanto !

Table of contents

Conclusion



Pipeline is here for a long time.
(Semi) structured knowledge can help in improving each stage.

Open issues / research directions

- **Models that are very parameter efficient (e.g., for mobile)** : do we really need $\mathcal{R}^{300}$? Can we (even) better compress model weights ?
- **Models that represent knowledge/context in latent space**
- **Labeling data**: we may have *all we need*. Further improvement may comes from collecting the correct data but it is expensive and error-prone :
  - we may further revisit **pseudo-labeling** and **self-training** tasks.
  - **F**urther solution for **zero and few-shots** settings.
- **Revise model practicability**: academic datasets are ("sometimes") biased. Some task we tag as solved may not be solved in practice (noisy data, more classes, user-generated contents, ...).

Actually anything that drive us from mimicking to **real understanding** (inc. common sense).

Questions ?

Tissier et al. (2019) – AAAI 2019 (pp. tbd)
Near-lossless Binarization of Word Embeddings

ElSahar et al. (2018) – NAACL-HLT 2018 (pp. 218–228)
Zero-Shot Question Generation from Knowledge Graphs for Unseen Predicates and Entity Types

Vougiouklis et al. (2018) – J. Web Semantics (52–53, pp. 1–15)
Neural Wikipedian: Generating Textual Summaries from Knowledge Base Triples

Kaffee et al. (2018) – NAACL-HLT 2018 (pp. 640–645)
Learning to Generate Wikipedia Summaries for Underserved Languages from Wikidata

Kaffee et al. (2018) – ESWC 2018 (pp. 319–334)
Mind the (Language) Gap: Generation of Multilingual Wikipedia Summaries from Wikidata for ArticlePlaceholders

ElSahar et al. (2018) – LREC 2018 (pp. 3448 – 3452)
T-REx: A Large Scale Alignment of Natural Language with Knowledge Base Triples

Tissier et al. (2017) – EMNLP 2017 (pp. 254 – 263)
Dict2vec : Learning Word Embeddings using Lexical Dictionaries

References I

- Bar-Haim, Roy et al. (2006). "The second pascal recognising textual entailment challenge". In: *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*. Vol. 6. 1. Venice, pp. 6–4.
- Bentivogli, Luisa et al. (2009). "The Fifth PASCAL Recognizing Textual Entailment Challenge.". In: *TAC*.
- Bojanowski, Piotr et al. (2017). "Enriching word vectors with subword information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146.
- Bordes, Antoine et al. (2015). "Large-scale simple question answering with memory networks". In: *arXiv preprint arXiv:1506.02075*.
- Dagan, Ido, Oren Glickman, and Bernardo Magnini (2005). "The PASCAL recognising textual entailment challenge". In: *Machine Learning Challenges Workshop*. Springer, pp. 177–190.

References II


Devlin, Jacob (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. https://nlp.stanford.edu/seminar/details/jdevlin.pdf. [Online; accessed 19-May-2019].


Devlin, Jacob et al. (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805*.


Dong, Li et al. (2017). "Learning to paraphrase for question answering". In: *arXiv preprint arXiv:1708.06022*.


Faruqui, Manaal et al. (2014). "Retrofitting word vectors to semantic lexicons". In: *arXiv preprint arXiv:1411.4166*.


Giampiccolo, Danilo et al. (2007). "The third pascal recognizing textual entailment challenge". In: *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*. Association for Computational Linguistics, pp. 1–9.

References III

- Levesque, Hector, Ernest Davis, and Leora Morgenstern (2012). "The winograd schema challenge". In: *Thirteenth International Conference on the Principles of Knowledge Representation and Reasoning*.
- Levy, Omer et al. (2015). "Do supervised distributional methods really learn lexical inference relations?". In: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 970–976.
- Luong, Minh-Thang et al. (2014). "Addressing the rare word problem in neural machine translation". In: *arXiv preprint arXiv:1410.8206*.
- Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

References IV


Ngonga Ngomo, Axel-Cyrille et al. (2013). "Sorry, i don't speak SPARQL: translating SPARQL queries into natural language". In: *Proceedings of the 22nd international conference on World Wide Web*. ACM, pp. 977–988.


Novikova, Jekaterina et al. (2017). "Why we need new evaluation metrics for NLG". In: *arXiv preprint arXiv:1707.06875*.


Peters, Matthew E. et al. (2018). "Deep contextualized word representations". In: *Proc. of NAACL*.


Ratner, Alexander et al. (2017). "Snorkel: Rapid training data creation with weak supervision". In: *Proceedings of the VLDB Endowment* 11.3, pp. 269–282.

References V

- Reddy, Sathish et al. (2017). "Generating natural language question-answer pairs from a knowledge graph using a rnn based question generation model". In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pp. 376–385.
- Serban, Iulian Vlad et al. (2016). "Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus". In: *arXiv preprint arXiv:1603.06807*.
- Wang, Alex et al. (2018). "Glue: A multi-task benchmark and analysis platform for natural language understanding". In: *arXiv preprint arXiv:1804.07461*.
- Wang, Alex et al. (2019). "SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems". In: *arXiv preprint arXiv:1905.00537*.

Table of contents

Remember

Some NLP models that are not "solved" [Devlin 2019]

- Models minimizing total training cost vs. accuracy on modern hardware
- **Models that are very parameter efficient (e.g., for mobile)**
- Models that represent knowledge/context in latent space
- Models that represent structured data (e.g., knowledge graph)
- Models that jointly represent vision and language

Limitations of real-valued vectors

- Require a **lot of space for storage**
  - Millions of words in vocabulary
  - 300 dimensions per vector
  - `float` usually requires 32 bits
  - $\Rightarrow$ 1.2 GB to store 1M vectors
- Not suitable for **embedded devices**
  - limited memory
  - low computing power

Binary representations

Solution: associate a m-bits vector to each word, i.e. embed real-value vectors into a Hamming hypercube of arbitrary dimensions ($\mathbb{R}^m \xhookrightarrow{dist} \mathbb{B}^n$).

Faster vector operations: Considerations

- – Cosine similarity requires $\mathcal{O}(n)$ additions and multiplications
- \+ Binary similarity requires a `XOR` and a `popcount()` operations that can be vectorized using CPU intrinsics (order of magnitude faster).

Small memory size

- – 32 bits$\times$300 dimensions $=$ 9600 bits per real-valued vectors
- \+ 128 or 256 bits per binary vectors
- \+ 16.1 MB to store 1M vectors of 128 bits (vs. 1.2 GB)
- \+ Computation can be done locally; no need of sending data to a computing server

Binary representations: Post-embedding scheme

Solution: associate a m-bits vector to each word, i.e. embed real-value vectors into a Hamming hypercube of arbitrary dimensions ($\mathbb{R}^m \xhookrightarrow{dist} \mathbb{B}^n$).

Binarize instead of training binary vectors

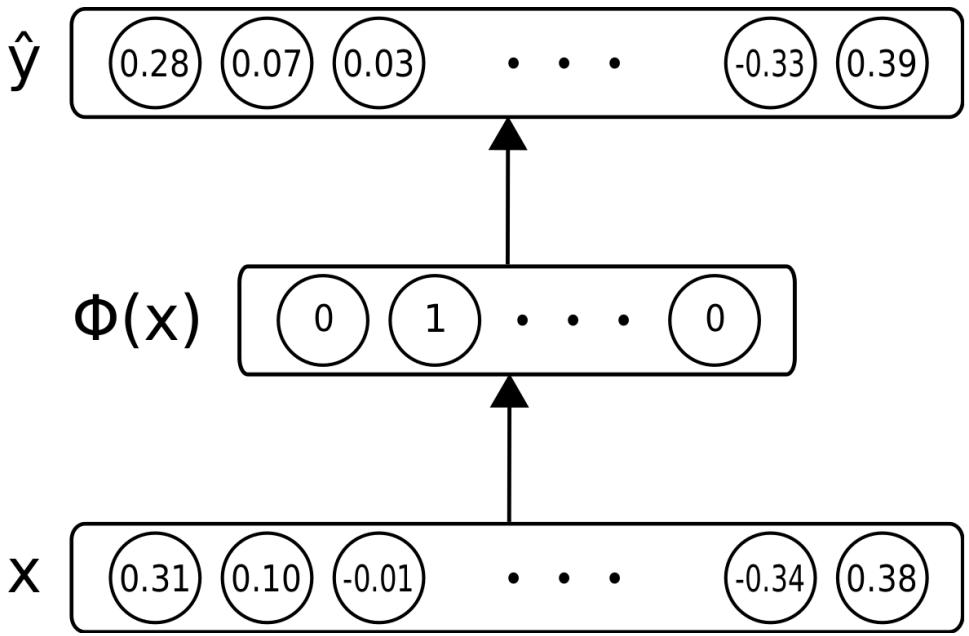
- Many pre-trained are already available
- Some are specific (subwords, dictionaries...). Need a **general method** that works for all types of architecture
- Preserve semantic and syntactic information

Why not a naive binarization ?

- Binary vectors size need to be **in adequacy with CPU registers size** (64, 128 or 256 bits) to benefit from hardware optimizations
- Pre-trained vectors have usually 300 dimensions

Auto-encoder architecture

- Use pre-trained vectors as input ($x \in \mathbb{R}^m$)
- Latent representation $\Phi(x)$ is binary ($\mathbb{B}^n$)
- Reconstruct a real-valued vector $\hat{y} \in \mathbb{R}^m$ given $\Phi(x) \in \mathbb{B}^n$
- How to back-propagate in the encoder, given the non-differentiability of the Heaviside function ?



Auto-encoder architecture

Let $W \in \mathbb{R}^{n \times m}$, $c \in \mathbb{R}^m$ and $x_i \in \mathbb{R}^m$

Encoding

- Heaviside function $h(x) = \begin{cases} 1 & if\ x \geq 0 \\ 0 & otherwise \end{cases}$
- $b_i = \Phi(x_i) = h(W \cdot x_i^T)$

\+ Can construct binary vectors of arbitrary size

Decoding

- $\hat{y}_i = tanh(W^T \cdot \Phi(x_i) + c)$
- Pre-trained embeddings have been cropped to be in [-1, 1] range

Objective Function

Reconstruction loss: minimize the distance between x_i and $\hat{y}_i$

$$\ell_{rec}(x_i) = \frac{1}{m} \sum_{k=0}^{m} (x_{i_k} - \hat{y}_{i_k})^2 \tag{4}$$

Regularization loss: minimize the correlation between each binary dimension

$$\ell_{reg} = \frac{1}{2} ||W^T W - I||^2 \tag{5}$$

Global Objective Function

$$\mathcal{L} = \sum_{x_i \in X} \ell_{rec}(x_i) + \lambda_{reg} \ell_{reg} \tag{6}$$

Word semantic similarity

- $BinarySimilarity(v_1, v_2) = \frac{n - HammingDistance(v_1, v_2)}{n}$
- dict2vec (2.3M), fasttext (1M), GloVe (400k)
- Best average results achieved with 256 bits (**37.5 times smaller**)

	dict2vec					fasttext					GloVe				
	raw	64	128	256	512	*raw*	64	128	256	512	*raw*	64	128	256	512
MEN	74.6					80.7					73.7				
bin	-	66.1	**71.3**	70.3	71.3	-	57.9	72.0	75.9	**76.3**	-	46.1	63.3	69.4	**72.7**
RW	50.5					53.8					41.2				
bin	-	36.5	42.0	**45.6**	45.6	-	36.8	44.7	**52.7**	52.7	-	25.1	34.3	**40.7**	40.2
SimLex	45.2					44.1					37.1				
bin	-	32.0	38.1	**44.8**	42.9	-	25.1	38.0	**44.6**	43.0	-	20.5	31.4	**37.2**	36.8
SimVerb	41.7					35.6					22.7				
bin	-	25.3	36.6	**38.4**	35.5	-	19.2	26.7	33.7	**35.1**	-	7.8	18.7	22.9	**23.0**
WS353	72.5					69.7					60.9				
bin	-	63.7	**71.6**	69.6	66.6	-	50.3	69.1	70.0	**70.3**	-	30.1	44.9	56.6	**60.3**

Table: Spearman's rank correlation scores on semantic similarity datasets for binary vectors (*bin*) of 64, 128, 256 and 512 bits and original vectors (*raw*).

Text classification

- Use the fasttext text classification model
- Initialize the network with the binary vectors
- 256/512 bits vectors are **on par/slightly outperforms** real vectors

	dict2vec					fasttext					GloVe				
	raw	64	128	256	512	*raw*	64	128	256	512	*raw*	64	128	256	512
AG-News	89.0					86.9					89.5				
bin	-	85.3	85.9	87.7	**87.8**	-	84.5	85.9	87.3	**87.7**	-	84.0	87.2	**88.5**	88.5
Amazon Full	47.5					49.0					47.1				
bin	-	39.9	43.9	46.8	**47.7**	-	39.0	43.9	47.9	**49.8**	-	37.4	42.6	46.7	**47.8**
DBpedia	97.6					95.0					97.2				
bin	-	94.1	96.1	97.0	**97.3**	-	91.7	95.1	96.6	**97.3**	-	90.9	95.0	96.8	**97.2**
Yahoo Ans	68.1					67.2					68.1				
bin	-	60.7	63.8	66.0	**66.8**	-	60.4	63.9	66.4	**67.8**	-	57.5	62.5	**66.4**	66.1

Table: Document classification accuracies for binary vectors (*bin*) of 64, 128, 256 and 512 bits and original vectors (*raw*).

Word semantic similarity and document classification

- Reconstruct 300-dimensional vectors from binary codes (W, c)
- Spearman's rank correlation score and document classification
- Sometimes reconstructed vectors are better than original (GloVe)

	dict2vec					fasttext					GloVe				
	raw	64	128	256	512	*raw*	64	128	256	512	*raw*	64	128	256	512
SimLex	45.2					44.1					37.1				
rec	-	30.4	37.9	**42.4**	39.3	-	19.2	30.0	**40.5**	34.0	-	19.6	19.1	34.2	**38.2**
WS353	72.5					69.7					60.9				
rec	-	61.4	**69.0**	67.4	58.8	-	36.5	53.6	**64.0**	53.6	-	26.5	42.2	56.5	**62.0**
Amazon Full	47.5					49.0					47.1				
rec	-	40.1	44.0	**46.8**	46.2	-	39.1	43.8	48.1	**48.5**	-	39.8	45.3	47.1	**47.3**
Yahoo Ans	68.1					67.2					68.1				
rec	-	60.8	63.8	65.9	**66.0**	-	60.0	62.9	66.3	**66.8**	-	58.4	64.3	66.7	**67.0**

Table: Word similarity scores and text classification accuracies vectors reconstructed (*rec*) from binary codes and original vectors scores (*raw*).

K-nearest neighbors search

- Find the k vectors with highest cosine/binary similarity

Execution time (ms)	128-bit	256-bit	512-bit	Real-valued
Top 1	2.87 (x34)	3.23 (x30)	4.28 (x23)	97.89
Top 10	2.89 (x34)	3.25 (x30)	4.29 (x23)	98.08
Loading vectors + Top 10	213 (x110)	310 (x75)	500 (x47)	23500

Table: Execution time (in ms) to run a top-K query on binary and real-valued vectors.

Recap

- Simple architecture to **binarize any real-valued word embeddings**
  - Optimization trick: Optimize decoder weights, apply transpose to encoder to avoid non-differentiability of *Heaviside*() function
  - Regularization trick: minimize correlation between binary dimension
- **Reduce the vector size by 37.5** with only 2% performance loss
- Top-k query **30 times faster** than with real-valued vectors

Open questions / research

- Quality trade-off w.r.t. other approximate kNN techniques in $\mathcal{R}^n$?
- The solution is agnostic to text classification and even NLP. How does it generalize to other application domains ?
- Embed representations built in other spaces than $\mathcal{R}^n$?
- Can we do ... even more compact binary codes ?

`https://github.com/tca19/near-lossless-binarization`