Word Vector Representation

Oct. 20th, 2020

Knowledge and Language Engineering Lab.,

Pohang University of Science and Technology

POSTECH
POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

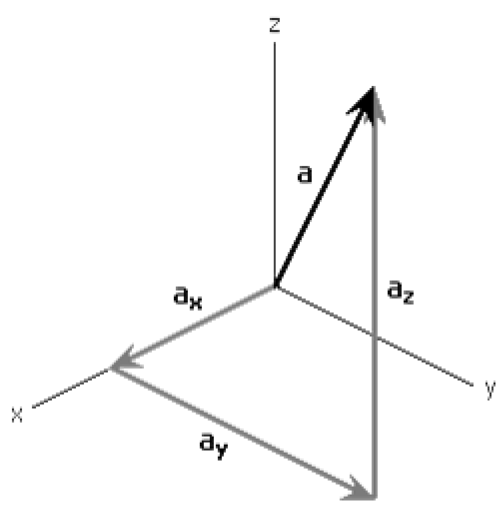
Contents

- Recapitulation
  - Vector Representation
  - Word2Vec (Mikolov et al., 2013)
  - GloVe (Pennington et al., 2014)
  - fastText (Bojanowski et al., 2017)
- Practice
  - Word2Vec
- Latest Trends

RECAPITULATION

Vector Representation

- Vector
  - An object that is supposed to represent its magnitude and direction at the same time.



- The arrow **a** is capable of signifying both its length (magnitude; ‘norm’) and angle (direction; ‘inner product’).

- $\boldsymbol{a} = (a_x, a_y, a_z)$.

Vector Representation

- **Language model (LM)**
  - A stochastic model for sequences of words.
  - LM can be used in all NLP tasks.

- Two ways to represent a word as a vector
  - discrete representation
  - continuous representation

Vector Representation

- Discrete Representation
  - A word is regarded as an autonomous entity.
  - Every word is represented as a '1-of-V' ('one-hot') vector of dimension $|V|$; sparse representation.

	Vector
Apple	[1, 0, 0, 0, ... , 0]
Banana	[0, 1, 0, 0, ... , 0]
Cherry	[0, 0, 1, 0, ... , 0]
Durian	[0, 0, 0, 1, ... , 0]
...	...

Vector Representation

- Discrete Representation
  - All word vectors are independent from one another.
  - → Redundant space (zero-valued)
  - → Intractability of a huge vocabulary
  - → Difficulty in estimating the similarity between two words

Vector Representation

- Continuous Representation
  - This system consists of dense matrices; ‘word embedding.’
  - Each dimension reflects a certain (uninterpreted) lexical property.

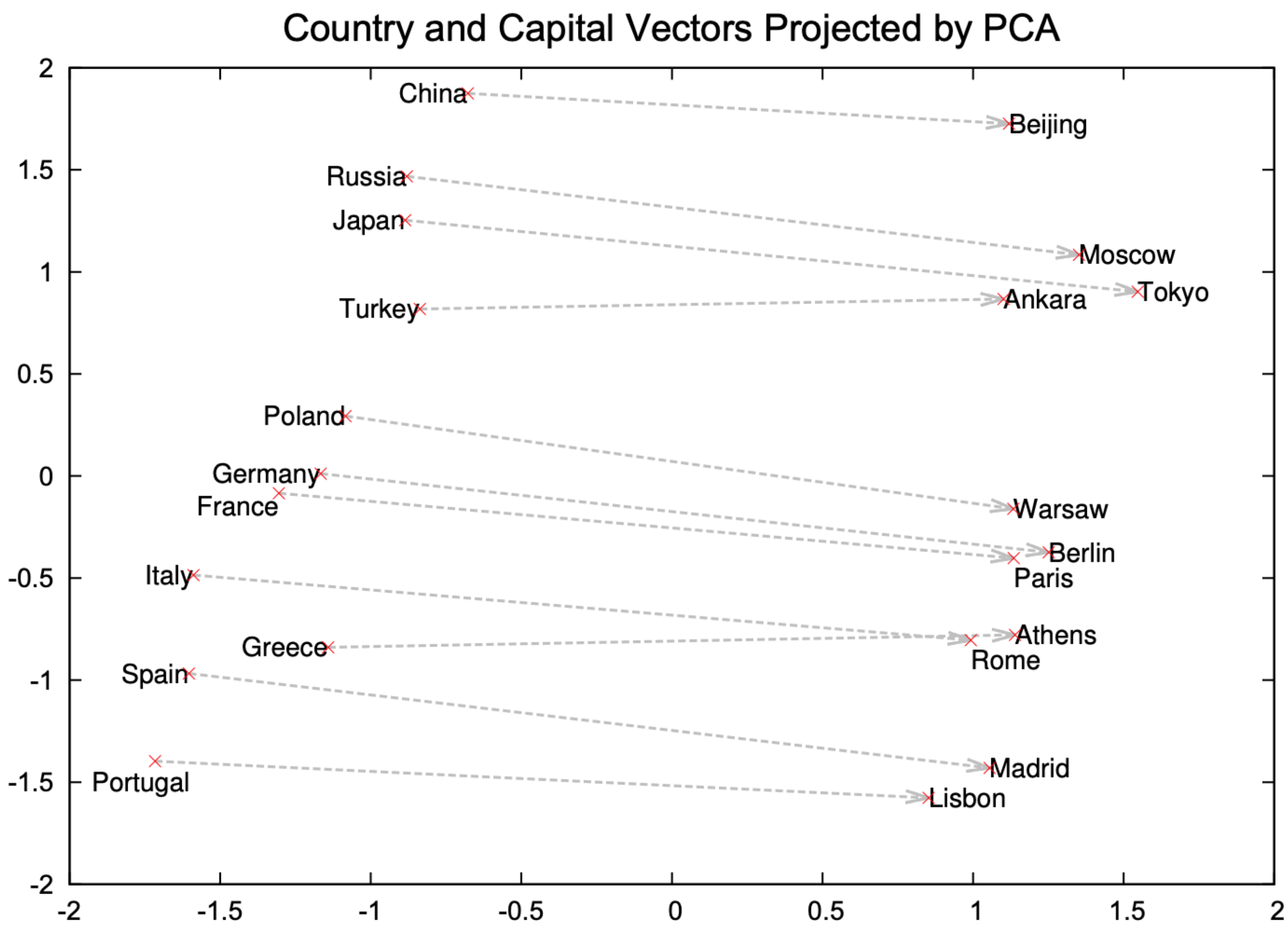
	Vector
Apple	[0.7, 0.5, 0.3, 0.1, ...]
Banana	[0.4, 0.1, 0.5, 0.9, ...]
Cherry	[0.6, 0.4, 0.7, 0.1, ...]
Durian	[0.2, 0.1, 0.6, 0.3, ...]
...	...

Vector Representation

- Distributional Hypothesis
  - “Words that are used and occur in the same context tend to purport similar meanings” (Harris 1954, as cited in “Distributional semantics," 2020, “Distributional hypothesis," para. 1)
  - “A word is characterized by the company it keeps” (Firth, 1968)
  - i.e. **co-occurrence**

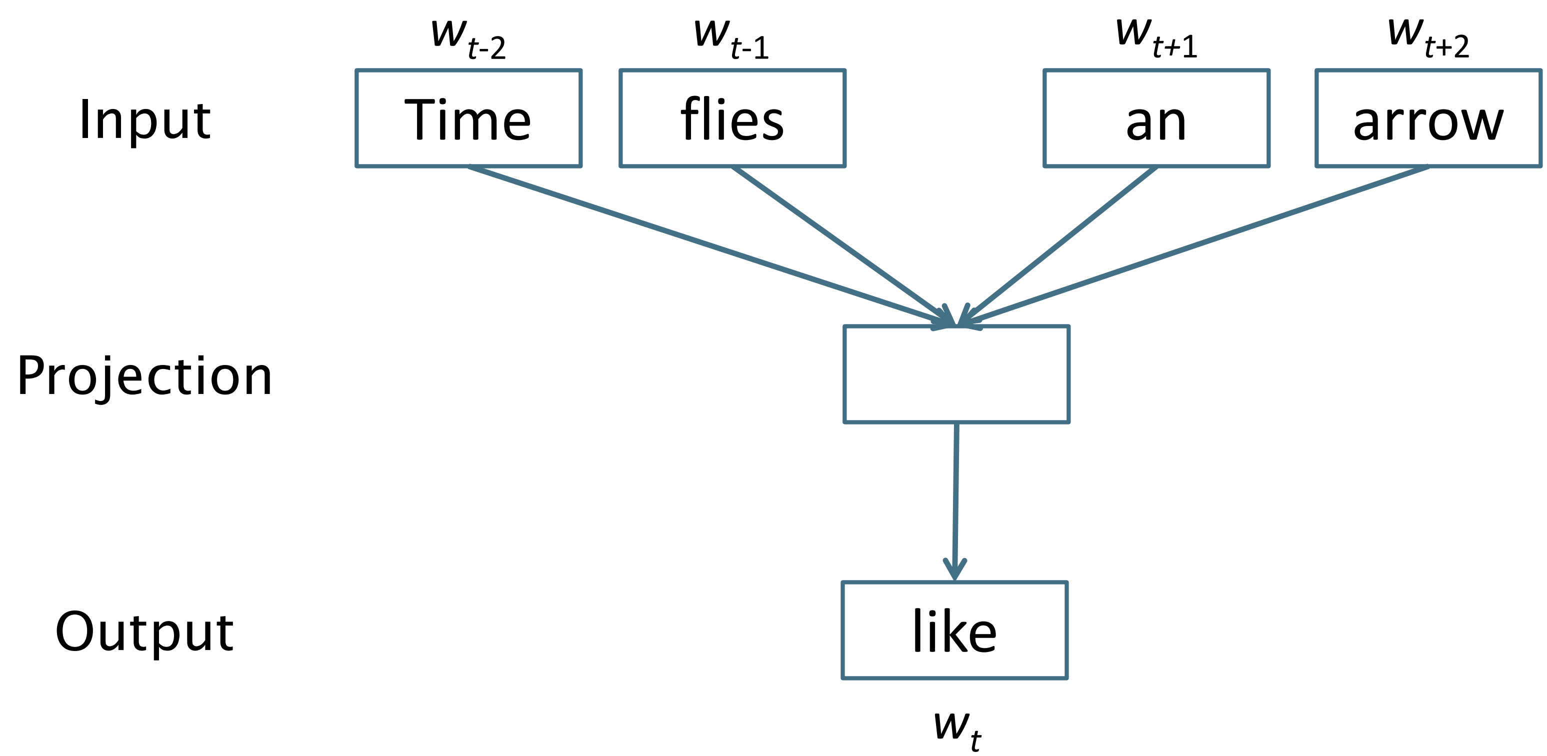
- Methods
  - Word2Vec (Mikolov et al., 2013)
  - GloVe (Pennington et al., 2014)
  - fastText (Bojanowski et al., 2017)

Word2Vec (Mikolov et al., 2013)



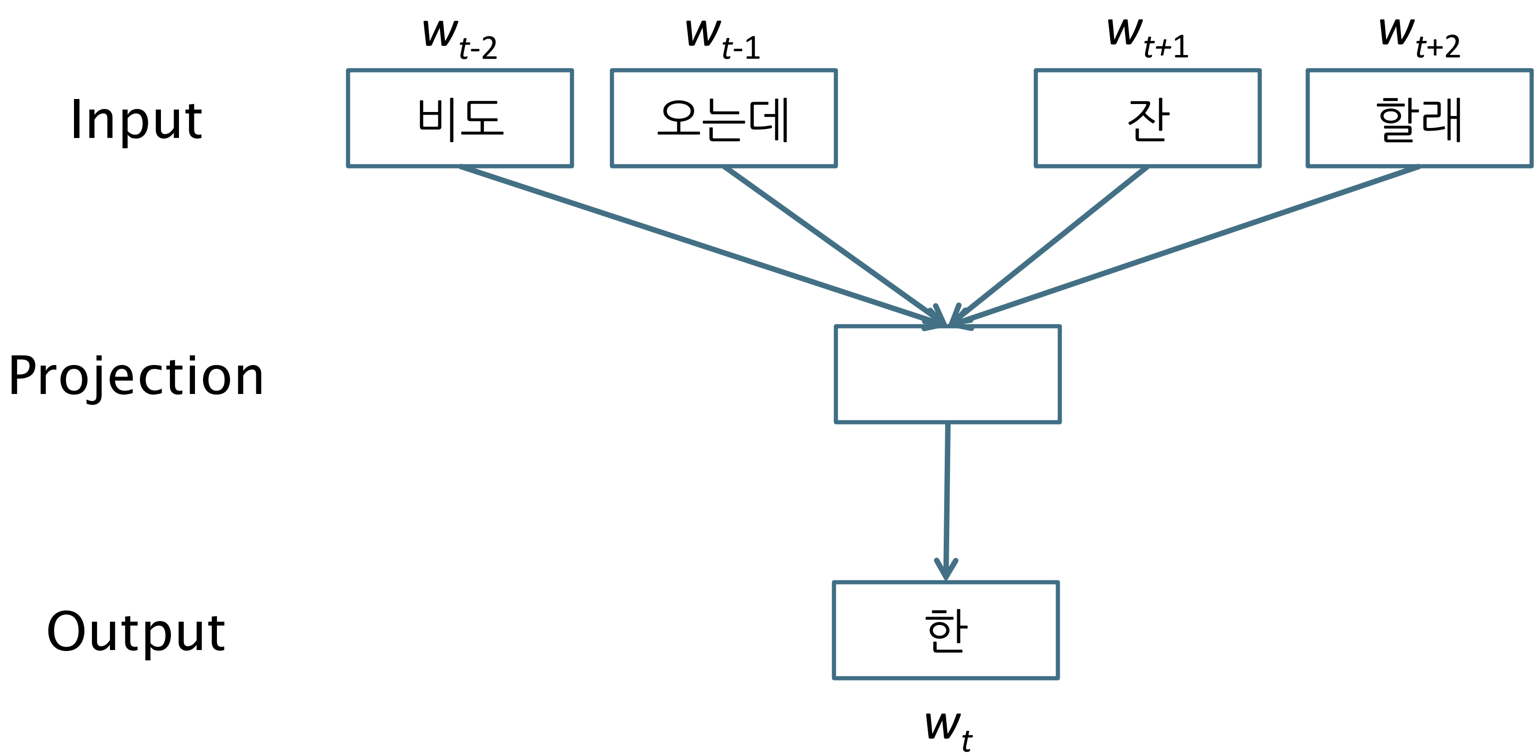
Word2Vec (Mikolov et al., 2013)

- Continuous Bag-of-Words Model
  - The model predicts the current word from the other words within a given range.



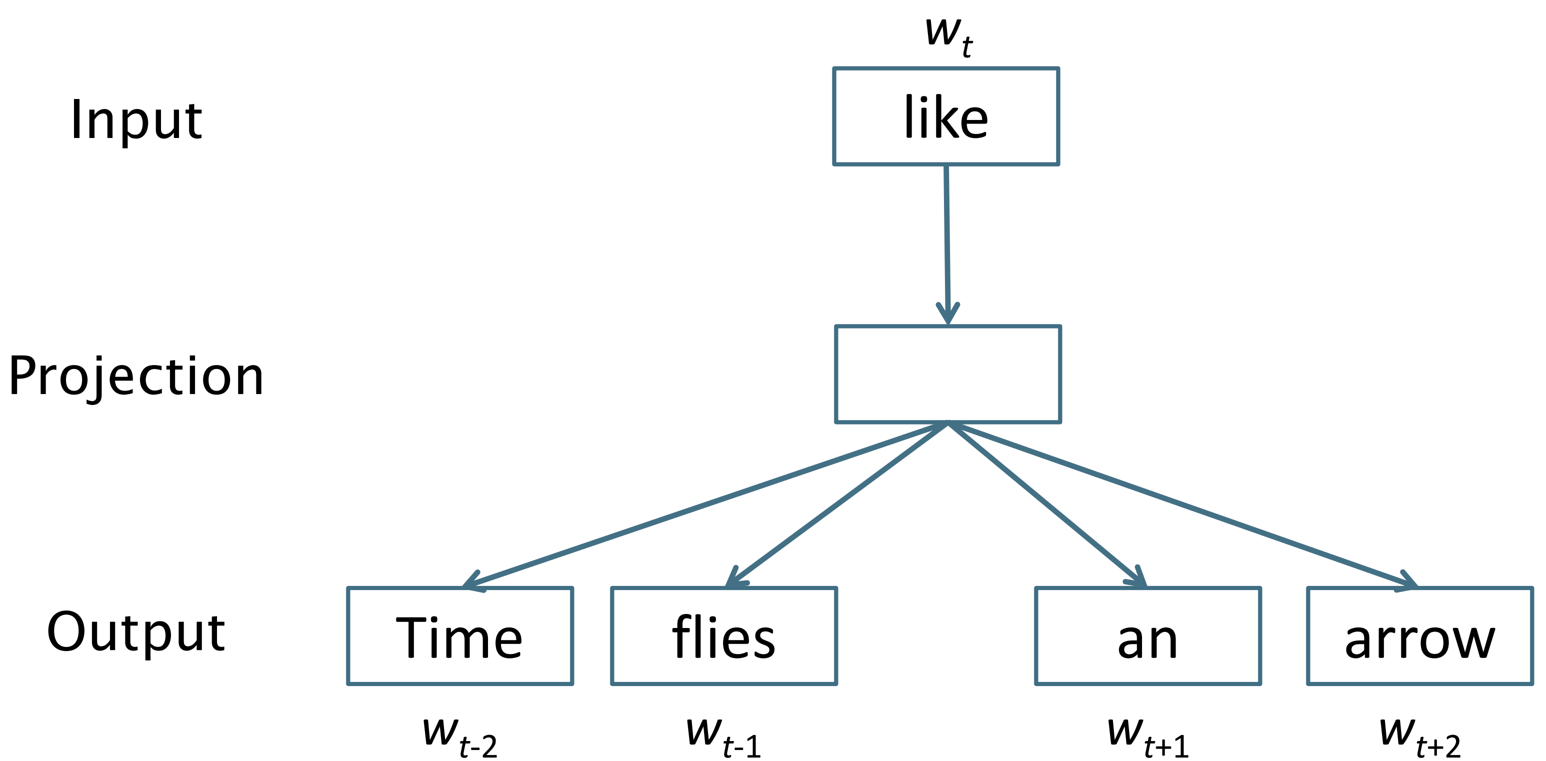
Word2Vec (Mikolov et al., 2013)

- Continuous Bag-of-Words Model
  - The model predicts the current word from the other words within a given range.



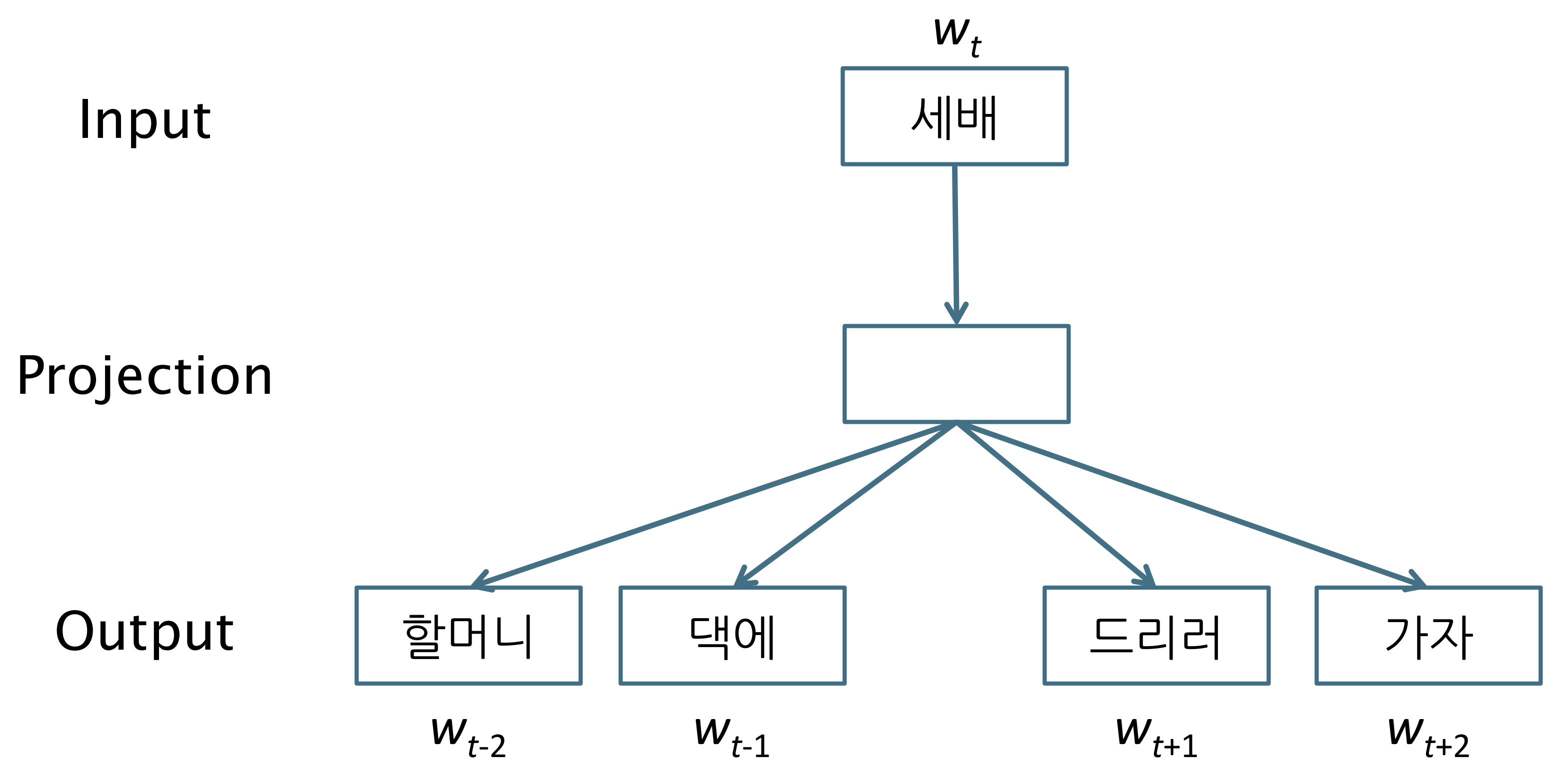
Word2Vec (Mikolov et al., 2013)

- Continuous Skip-gram Model
  - The model predicts the other words from the current word



Word2Vec (Mikolov et al., 2013)

- Continuous Skip-gram Model
  - The model predicts the other words from the current word



GloVe (Pennington et al., 2014)

- **Glo**bal **Ve**ctors for Word Representation
- *"Word2Vec methods poorly utilize the statistics of the corpus since they train on separate local context windows instead of on global co-occurrence counts."*
- The model is based on the co-occurrence matrix.
  - *A "global skip-gram" model.*

	$k =$ solid	$k =$ gas	$k =$ water	$k =$ fasion
$\mathrm{P}(k\|\text{ice})$	0.00019	0.000066	0.003	0.000017
$\mathrm{P}(k\|\text{steam})$	0.000022	0.00078	0.0022	0.000018
$\mathrm{P}(k\|\text{ice}) / \mathrm{P}(k\|\text{steam})$	8.9	0.085	1.36	0.96

fastText (Bojanowski et al., 2017)

- We can handle rare words by decomposing them into ‘subwords’.
- A pre-trained Korean embedding is provided at
  - https://fasttext.cc/docs/en/crawl-vectors.html

```
u-n_r-e-l-a_t-e_d
u-n re-l_a-t-e_d
u-n re_l-at-e_d
un re-l-at-e-d
un re_l-at-ed
un re-lat-ed
un relat_ed
```

```
u-n-r-e-l-a_t-e-d
u_n re_l-a-t-e-d
u_n re-l-at-e-d
u_n re-l-ate_d
u_n rel-ate-d
u_n relate_d
```

```
u-n_r_e_l-a-t-e-d
u-n-r_e-l-at-e-d
u-n-r_e-l_at_ed
un-r-e-l-at-ed
un re-l_at-ed
un re-l-ated
un rel_ated
```

fastText (Bojanowski et al., 2017)

- Word2Vec taking the subword level into account
  - Word2Vec's score function $s(w_t, w_c)$ is the inner product of the word and its context, but fastText uses

$$s(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^\top \mathbf{v}_c$$

    - , where z_g is a n-gram subword vector and v_c is the context vector.

PRACTICE

Gensim

- A free Python library for statistical semantics
  - https://radimrehurek.com/gensim/index.html
- Install Gensim
  - `sudo pip install --upgrade gensim`
  - conda install gensim
  - For more information
    - https://radimrehurek.com/gensim/install.html

Practice

1. Prepare the corpus to train
2. Train a Word2Vec model
3. Load the trained model
4. Score the similarity between words
5. Find the word farthest away from the mean
6. Find the top-N most similar words
7. Find the top-N most similar words with combination of words

Practice

- Step 1. Prepare the corpus to train
  - Korean news corpus
    - Crawled from online news sites
    - 430 k sentences, 160 k morphemes
    - Morphologically analyzed (No POS Tags)
    - Word frequencies

Frequency	≥ 1000	≥ 700	≥ 500	≥ 300	≥ 100
Unique Words	1612	2175	2891	4250	9196

Practice

- Step 2. Train a Word2Vec model
  - `model = gensim.models.Word2Vec(size=150, window=5, sg=1, min_count=5, worker=20)`
    - `size`: the dimension of word vector, default = 100
    - `window`: the size of word window, default = 5
    - `sg`: 0 – CBOW / 1 – skip-gram, default = 0
    - `min_count`: the threshold of word frequency, default = 5
    - `worker`: the number of thread for training, default = 1

Practice

- Step 2. Train a Word2Vec model
  - `model.build_vocab(sentences)`
    - `sentences`: the texts for training
  - `model.train(sentences, total_words=len(model.wv.vocab), epochs=model.epochs)`
  - `model.save($model_name)`
    - `$model_name`: the file name of the saved model

Practice

- Step 3. Load the trained model
  - `model = gensim.models.Word2Vec.load($model_path)`
    - `$model_path`: the location of the trained model

Practice

- Step 4. Score the similarity between words
  - `model.wv.similarity(`*`word1, word2`*`)`
    - Score the similarity of `word1` and `word2`

- Examples
  - 한국 - 북한: 0.995
  - 노트북 - 컴퓨터: 0.994
  - 일본 - 도쿄: 0.987
  - 자동차 - 휘발유: 0.982
  - 임상실험 - 신약: 0.933
  - 파인애플 - 피자: 0.147

Practice

- Step 5. Find the word that is farthest away from the mean of the given words including itself.
  - `model.wv.doesnt_match(`*`word_list`*`)`
    - Returns the word that is farthest away from the mean of `word_list`
- Examples
  - 소프트웨어 하드웨어 컴퓨터 치약 → 치약
  - 국회 정부 정책 창문 → 창문
  - 버스 지하철 비행기 자가용 → 자가용

Practice

- Step 6. Find the top-N most similar words
  - `model.wv.most_similar(positive=[word])`
  - Print 10 most similar words

Practice

- Step 7. Find the top-N most similar words with a combination of words

  - ```
    model.wv.most_similar(positive=[words],
                          negative=[words], topn=1)
    ```

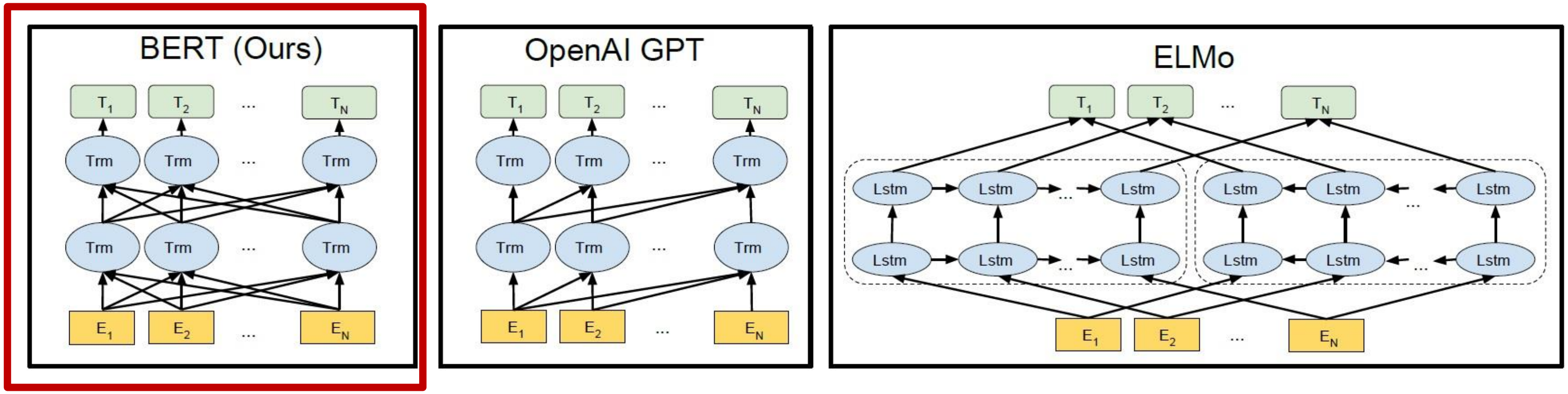
    - positive/negative: (pos/neg) words for calculation
    - topn: the number of the most similar words

- Example
  - Find the most similar word with the result of (a – b + c)
  - 대통령 - 한국 + 미국 → 부시
  - https://word2vec.kr/search/
    ```

LATEST TRENDS

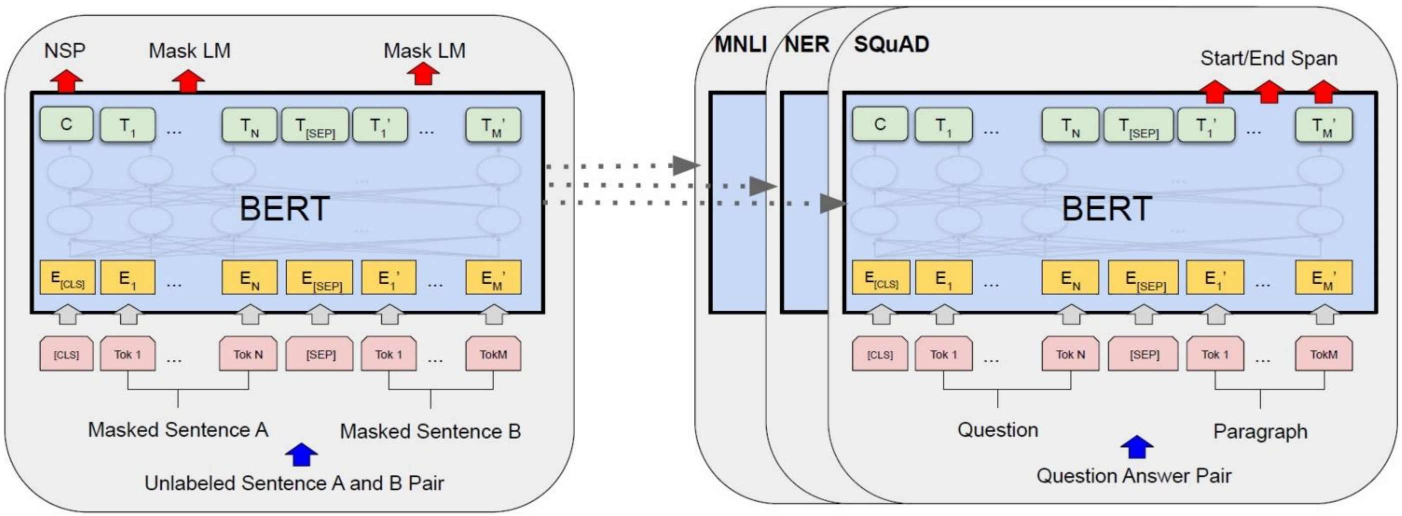
BERT (Devlin et al., 2019)

- Contextualized Word Representation
  - Pre-trained on huge unlabeled corpora.
  - Using Transformer's encoder (multi-head self-attention).
  - https://github.com/google-research/bert



Usage of BERT

- Transferring Features
  - Using some of layer values as features for down-stream tasks.
- Fine-tuning
  - Tuning pre-trained parameters with a touch of down-stream tasks.



Pre-training

Fine-Tuning

Popular variants of BERT

- XLNet (Yang et al., 2019)
  - https://github.com/zihangdai/xlnet
- XLM (Conneau & Lample, 2019)
  - https://github.com/pytorch/fairseq/tree/master/examples/cross_lingual_language_model
- RoBERTa (Liu et al., 2019)
  - https://github.com/pytorch/fairseq/tree/master/examples/roberta
- BART (Lewis et al., 2020)
  - https://github.com/pytorch/fairseq/tree/master/examples/bart

Q & A

References

- Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and Their Compositionality, In *Advances in Neural Information Processing Systems 26*, Red Hook, NY, USA, Curran Associates Inc. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf
- Pennington, J., Socher, R., & Manning, C. (2014). GloVe: Global Vectors for Word Representation, In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Association for Computational Linguistics. https://doi.org/10.3115/v1/D14-1162
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, (5), 135–146. https://doi.org/10.1162/tacl_a_00051
- contributors, W. (2020). Distributional semantics - Wikipedia [Online; accessed 18-October-2020]. https://en.wikipedia.org/wiki/Distributional_semantics
- Harris, Z. S. (1954). Distributional Structure. *WORD*, *2-3*(10), 146–162. https://doi.org/10.1080/00437956.1954.11659520
- Firth, J. R. (1968). Studies in Linguistic Analysis (F. R. Palmer, Ed.). In F. R. Palmer (Ed.), *Selected Papers of J.R. Firth 1952-1959*. Original work published 1957. London, Longman.

References

- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (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)*, Minneapolis, Minnesota, Association for Computational Linguistics.
- Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., & Le, Q. V. (2019). XLNet: Generalized Autoregressive Pretraining for Language Understanding, In *Advances in Neural Information Processing Systems 32*, Red Hook, NY, USA, Curran Associates, Inc. http://papers.nips.cc/paper/8812-xlnet-generalized-autoregressive-pretraining-for-language-understanding.pdf
- Conneau, A., & Lample, G. (2019). Cross-lingual Language Model Pretraining, In *Advances in Neural Information Processing Systems 32*, Red Hook, NY, USA, Curran Associates, Inc. http://papers.nips.cc/paper/8928-cross-lingual-language-model-pretraining.pdf
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. https://arxiv.org/abs/1907.11692
- Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension, In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.703