Sentence Textual Similarity: Model Evolution Overview

Shuyue Jia

Ph.D. Student,

Boston University

October 24th, 2023

Dependable Computing Laboratory,
Department of Electrical and Computer Engineering,
Boston University

Outline

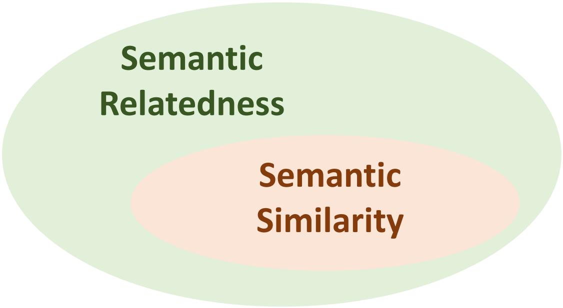
- Background and Preliminary
- Distance Measurement
- Model Evolution Overview
- Matrix-based Methods
- Alignment/Attention-based Methods
- Word Distance-based Methods
- Sentence Embedding-based Methods
- Evaluation Metrics
- Benchmark Databases
- Takeaways

Dependable Computing Laboratory,
Department of Electrical and Computer Engineering,
Boston University

Part 1 – Background of *Semantic Textual Similarity (STS)*

- **What**: quantitatively measure of **semantic equivalence** between two blocks of texts, *i.e.*, sentences
- **How**: compute **semantic scores** between texts, measured by **semantic distance**
- **Intuition**: **rich interaction structures** in the text-matching process (words, phrases, whole sentences)
- **Challenge**: lack of large-scale **labeled** datasets (**sentence pairs** ⟺ **labeled semantic similarity scores**)
- **Semantic Relatedness** *vs.* **Semantic Similarity**:

  Broader perspective analyzing the shared semantic properties of two texts (*e.g.*, “coffee” and “mug”)



Credits: Chandrasekaran *et al.*, Evolution of Semantic Similarity—A Survey, In ACM Computing Survey’21.

Part 1 – Preliminary of *Word Embedding*

- **Bag-of-words Model (BoW)**: a model of text represented as an **unordered collection of words**, mainly used to **calculate frequencies of words** in different documents [1]
  - "John", "likes", "to", "watch", "movies", "Mary", "likes", "movies", "too"
  - BoW1 = {"John":1, "likes":2, "to":1, "watch":1, "movies":2, "Mary":1, "too":1}
- **Word Embedding**: **continuous vector representations** of words **that encode the meaning of the word** in such a way that words that are closer in the vector space are expected to be similar in meaning [2]
- **Distributional Hypothesis**: two words tend to be **semantically close** if they occur in **similar contexts** [3]
- **Word2Vec** [2]:
  - Continuously sliding bag-of-words (**CBOW**) – **Neighbor words** predict **center word**
  - Continuously sliding **skip-gram** – **Center word** predicts **neighbor words**

Credits: [1] Li *et al.*, Sentence Similarity Based on Semantic Nets and Corpus Statistics, In IEEE T-KDE'06.
[2] Mikolov *et al.*, Efficient Estimation of Word Representations in Vector Space, In ICLR'13. [3] Firth *et al.*, A Synopsis of Linguistic Theory, In Studies in Linguistic Analysis'1930.

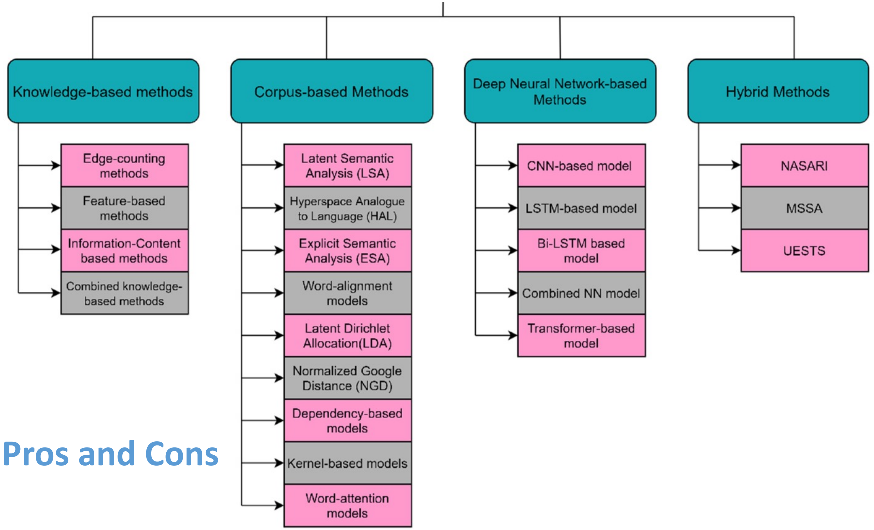
Part 1 — Preliminary of *Word Embedding*

- **FastText**: **average of character n-grams**, can provide embeddings for Out-of-vocabulary (OOV) words [1]
  - "equal" ← "eq", "equ", "qua", "ual", "al"
- **GloVe**: **word-word co-occurrences** within context windows ⟹ **ratios of co-occurrence probabilities** [2]
  - $P_{ij} = P(i|j) = X_{ij} / \sum_k X_{ik} \Longrightarrow$ the probability that word j appear in the context of word i
- **Skip-Thought Embeddings**: **Center sentence** predicts **neighbor sentences** [3]
- **Autoencoder BERT Embeddings**: **Masked Language Modeling** and **Next Sentence Prediction** [4]
  - "[CLS] The man went to the store. [SEP] He bought a gallon of milk."
- **Universal Sentence Encoder**: **sentence embedding** by **element-wise sum** and **length normalization** [5]

Credits: [1] Bojanowski *et al.*, Enriching Word Vectors with Subword Information, In TACL'17.
[2] Pennington *et al.*, GloVe: Global Vectors for Word Representation, In EMNLP'14.
[3] Kiros *et al.*, Skip-Thought Vectors, In NeurIPS'15.
[4] Devlin *et al.*, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, In NAACL'19.
[5] Cer *et al.*, Universal Sentence Encoder, In arXiv'18.

Part 1 – Preliminary of *Knowledge and Statistics Methods*

- **Faced with an *embarras de richesse***
- Bag-of-words Model (BoW) [1]
- Term Frequency-Inverse Document Frequency (TF-IDF) [2,3]
- Latent Semantic Indexing (LSI) [4]
- Latent Dirichlet Allocation (LDA) [5]
- **Traditional Works** mostly focus on **Word Similarity** [6]
- **Please check [6] to know more about Previous Works** and **Pros and Cons**



Credits:
[1] Li *et al.*, Sentence Similarity Based on Semantic Nets and Corpus Statistics, In IEEE T-KDE'06.
[2] Luhn *et al.*, A Statistical Approach to Mechanized Encoding and Searching of Literary Information, In IBM Journal of Research and Development'1957.
[3] Jones *et al.*, A Statistical Interpretation of Term Specificity and its Application in Retrieval, In Document Retrieval Systems'1988.
[4] Deerwester *et al.*, Indexing by Latent Semantic Analysis, In Journal of the American Society for Information Science'1990.
[5] Blei *et al.*, Latent Dirichlet Allocation, In JMLR'03.
[6] Chandrasekaran *et al.*, Evolution of Semantic Similarity—A Survey, In ACM Computing Survey'21.

Part 1 – Preliminary of *Baseline*

- **Avg. GloVe Embeddings**: average of GloVe embeddings [1]
- **Avg. Skip-Thought Embeddings**: average of word embeddings produced by Skip-Thought vectors [2]
- **InferSent**: a Siamese BiLSTM network with max-pooling over the output on NLI datasets [3]
- **Avg. BERT Embeddings**: average of word embeddings produced by BERT [4]
- **BERT [CLS]**: scores based on the vector representation of the special token [CLS] in BERT [4]
- **BERTScore**: the similarity of sentences as a sum of cosine similarities between tokens' embeddings [5]
- **BLEURT**: based on BERT and captures similarities by fine-tuning the model [6]
- **DPR**: two unique BERT encoders and the model weights are optimized to maximize the dot product [7]
- **Universal Sentence Encoder**: encoding sentences into their corresponding embeddings [8]
- **Sentence-BERT**: BERT + Siamese structure to derive sentence embeddings + compared through cosine similarity [9]

Credits: [1] Pennington *et al.*, GloVe: Global Vectors for Word Representation, In EMNLP'14. [2] Kiros *et al.*, Skip-Thought Vectors, In TACL'22.
[3] Conneau *et al.*, Supervised Learning of Universal Sentence Representations from Natural Language Inference Data, In EMNLP'17.
[4] Devlin *et al.*, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, In NAACL'19.
[5] Zhang *et al.*, BERTScore: Evaluating Text Generation with BERT, In ICLR'20. [6] Sellam *et al.*, BLEURT: Learning Robust Metrics for Text Generation, In ACL'20.
[7] Karpukhin *et al.*, Dense Passage Retrieval for Open-Domain Question Answering, In EMNLP'20.
[8] Cer *et al.*, Universal Sentence Encoder, In arXiv'18. [9] Reimers *et al.*, Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, In EMNLP'19.

Part 2 – Distance Measurement – *Spatial Metrics*

- **Cosine Similarity** – words co-occur → small angle → large cosine

$$\mathrm{Cos}(w_1, w_2) = \frac{\sum_{w \in C(w_1) \cup C(w_2)} P(w|w_1) \times P(w|w_2)}{\sqrt{\sum_{w \in C(w_1)} P(w|w_1)^2} \times \sqrt{\sum_{C(w_2)} P(w|w_2)^2}}.$$

- **Manhattan Distance** or L_1 **Norm**

$$\mathrm{L}_1(w_1, w_2) = \sum_{w \in C(w_1) \cup C(w_2)} |P(w|w_1) - P(w|w_2)|.$$

- **Euclidean Distance** or L_2 **Norm**

$$\mathrm{L}_2(w_1, w_2) = \sqrt{\sum_{w \in C(w_1) \cup C(w_2)} \left(P(w|w_1) - P(w|w_2)\right)^2}.$$

Semantic distance: a measure of how close or distant the meanings of two units of language are.

Notions:

$C(w)$: the set of words that co-occur (within a certain window) with the word w in a corpus

$P(w|w_1)$: conditional probability of the co-occurring words given the target words is used as the strength of association

Credits: Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

Part 2 – Distance Measurement – *Mutual Information*

- **Hindle**

$$\mathrm{Hin}(w_1, w_2) = \sum_{w \in C(w)} \begin{cases} \min\big(I(w, w_1), I(w, w_2)\big), & \text{if both } I(w, w_1) \text{ and } I(w, w_2) \; > \; 0; \\ \left|\max\big(I(w, w_1), I(w, w_2)\big)\right|, & \text{if both } I(w, w_1) \text{ and } I(w, w_2) \; < \; 0; \\ 0, & \text{otherwise.} \end{cases}$$

$I(w_1, w_2)$ is the Pointwise Mutual Information (PMI) between w_1 and w_2. Using the minimum of the two PMIs captures the similarity of the occurred words w and two words w_1 and w_2.

- **Lin**

$$\mathrm{Lin}(w_1, w_2) = \frac{\sum_{(r,w) \in T(w_1) \cap T(w_2)} \big(I(w_1, r, w) + I(w_2, r, w)\big)}{\sum_{(r,w') \in T(w_1)} \big(I(w_1, r, w')\big) + \sum_{(r,w'') \in T(w_2)} \big(I(w_2, r, w'')\big)},$$

where the word w_1 is related to w by the syntactic relation r, and $T(w_1)$ is the set of all word pairs (r, w) such as pos. I.

Credits: Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

Part 2 — Distance Measurement — *Relative Entropy*

- **Kullback-Leibler Divergence/Distance (KLD)/Relative Entropy — Common Occurrence**

$$\mathrm{KLD}(w_1, w_2) = D\big(P(w|w_1)||P(w|w_2)\big) = \sum_{w \in C(w_1) \cup C(w_2)} P(w|w_1) \log \frac{P(w|w_1)}{P(w|w_2)}.$$

The more $P(w|w_1)$ and $P(w|w_2)$ are similar, the more w_1 and w_2 are semantically similar.

- **Kullback-Leibler Divergence (KLD) — Absolute**

$$\sum_{w \in C(w_1) \cup C(w_2)} P(w|w_1) \left| \log \frac{P(w|w_1)}{P(w|w_2)} \right|.$$

- **Kullback-Leibler Divergence (KLD) — Average**

$$\frac{1}{2} \sum_{w \in C(w_1) \cup C(w_2)} \big(P(w|w_1) - P(w|w_2)\big) \log \frac{P(w|w_1)}{P(w|w_2)}.$$

- **Kullback-Leibler Divergence (KLD) — Maximum**

$$\max\big(\mathrm{KLD}(w_1, w_2), \mathrm{KLD}(w_2, w_1)\big).$$

Credits: Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

Part 2 – Distance Measurement – *Relative Entropy*

- **α-skew Divergence (ASD)**

  A slight modification of the KLD that obviates the need for smoothed probabilities

$$\mathrm{ASD}(w_1, w_2) = \sum_{w \in C(w_1) \cup C(w_2)} P(w|w_1) \log \frac{P(w|w_1)}{\alpha P(w|w_2) + (1-\alpha) P(w|w_1)},$$

where α is usually set to 0.99. Better estimate word co-occurrence probabilities than KLD.

- **Jensen-Shannon Divergence (JSD)/total divergence to the average/information radius**

  A relative entropy–based measure that overcomes the problem of asymmetry in KLD

$$\mathrm{JSD}(w_1, w_2) = \sum\nolimits_{w \in C(w_1) \cup C(w_2)} \left(P(w|w_1) \log \frac{P(w|w_1)}{\frac{1}{2}\big(P(w|w_1) + P(w|w_2)\big)} + P(w|w_2) \log \frac{P(w|w_2)}{\frac{1}{2}\big(P(w|w_1) + P(w|w_2)\big)} \right).$$

Credits: Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

Part 2 — Distance Measurement

- **Co-occurrence Retrieval Models (CRM)**

$$\mathrm{CRM}(w_1, w_2) = \gamma\left[\frac{2\times P\times R}{P+R}\right] + (1-\gamma)\big[\beta[P] + (1-\beta)[R]\big].$$

- **Dice Coefficient**

$$\mathrm{Dice}(w_1, w_2) = \frac{2\times \sum_{w\in C(w_1)\cup C(w_2)} \min\big(P(w|w_1), P(w|w_2)\big)}{\sum_{w\in C(w_1)} P(w|w_1) + \sum_{w\in C(w_2)} P(w|w_2)}.$$

- **Division Measure**

$$\mathrm{Division}(w_1, w_2) = \sum_{w\in C(w_1)\cup C(w_2)} \left|\log\frac{P(w|w_1)}{P(w|w_2)}\right|.$$

- **Jaccard**

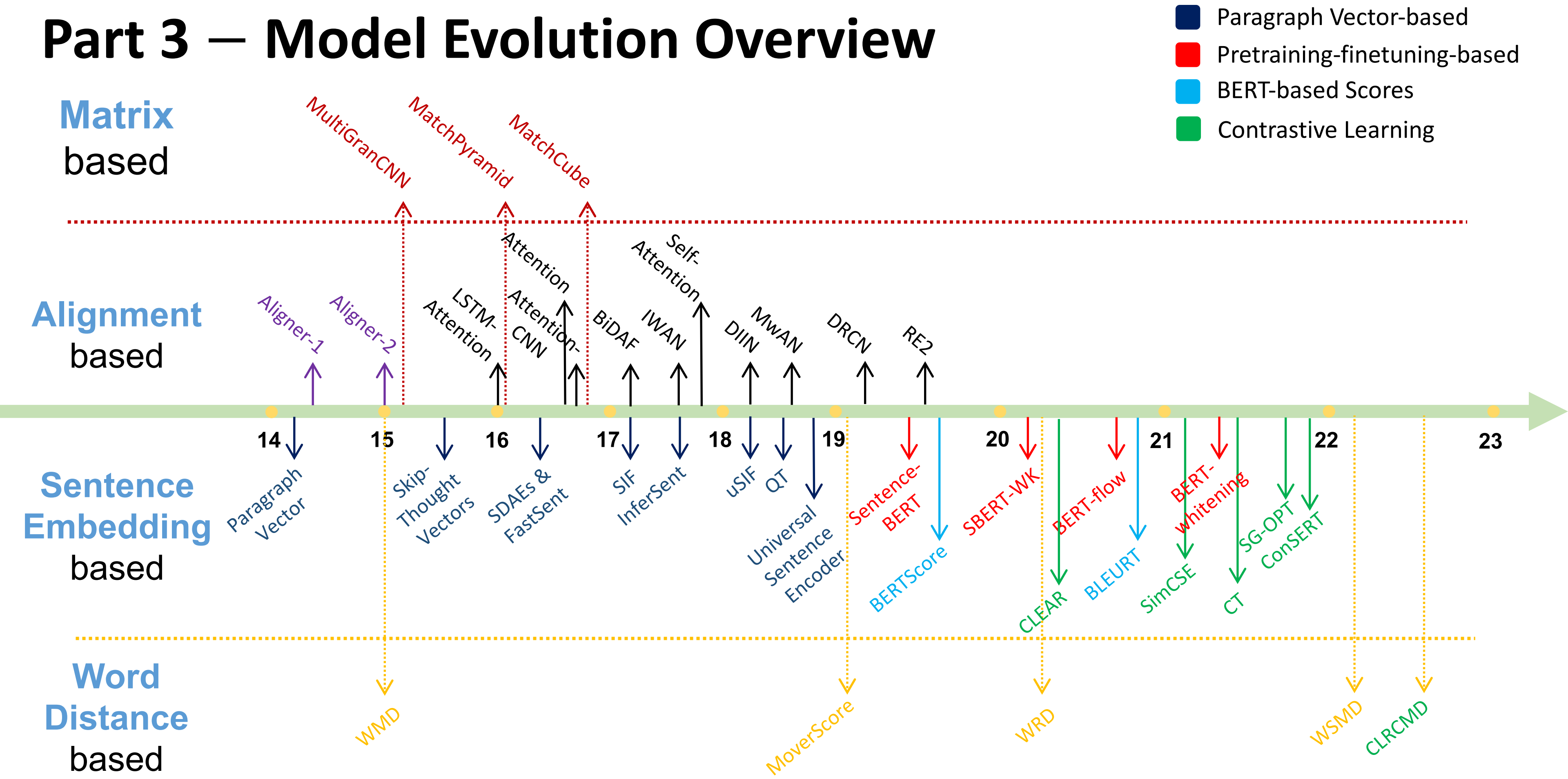
$$\mathrm{Jaccard}(w_1, w_2) = \frac{\sum_{w\in C(w_1)\cup C(w_2)} \min\big(P(w|w_1), P(w|w_2)\big)}{\sum_{w\in C(w_1)\cup C(w_2)} \max\big(P(w|w_1), P(w|w_2)\big)}.$$

- **Product Measure**

$$\mathrm{Product}(w_1, w_2) = \sum_{w\in C(w_1)\cup C(w_2)} \frac{P(w|w_1)\times P(w|w_2)}{\left(\frac{1}{2}P(w|w_1) + P(w|w_2)\right)^2}.$$

Credits:
Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

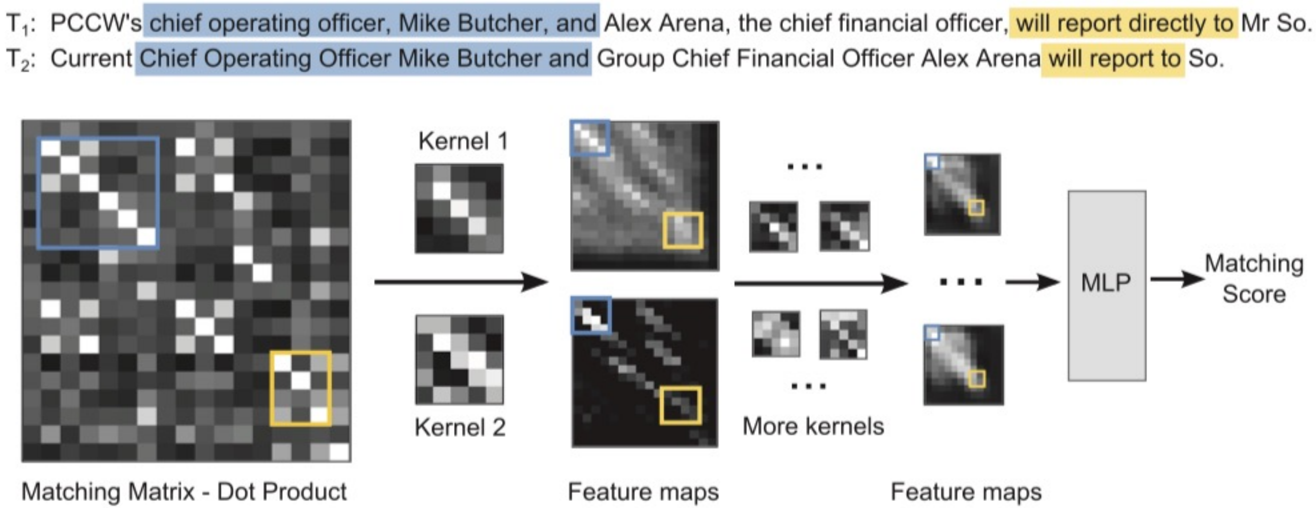
Part 3 — Model Evolution Overview



Part 4 — Matrix-based Methods

➢ **Key Idea**: construct a **similarity matrix** between two sentences, each element of which represents the similarity between the two corresponding units in two sentences. Then, the matrix is aggregated in different ways to induce the **final similarity score** [1]

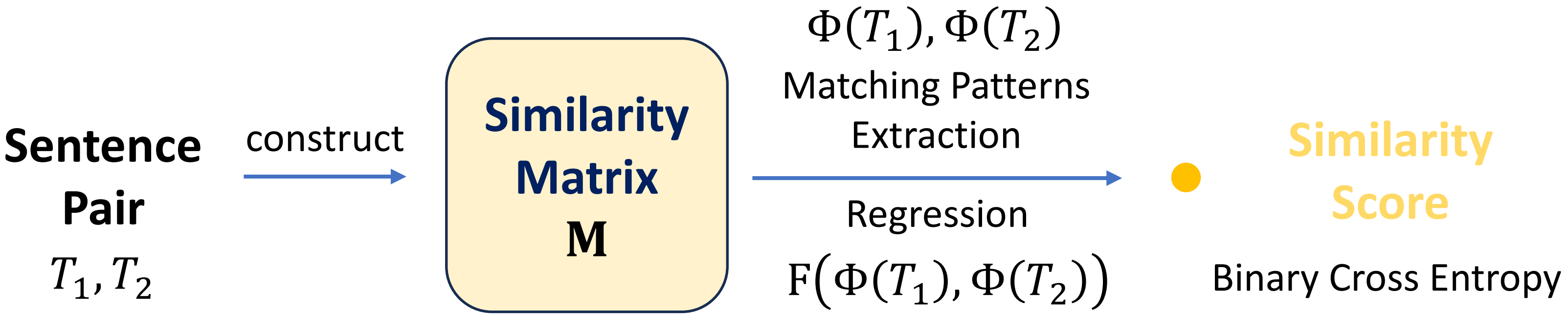
Semantic Correspondence



Credits: [1] Sun *et al.*, Sentence Similarity Based on Contexts, In TACL'22. [2] Pang *et al.*, Text Matching as Image Recognition, In AAAI'16.

Part 4 — Matrix-based Methods

- **Key Idea**: construct a **similarity matrix** between two sentences, each element of which represents the similarity between the two corresponding units in two sentences. Then, the matrix is aggregated in different ways to induce the **final similarity score** [1]



- **Similarity Matrix** (Matching Matrix) $\mathbf{M}$: similarity between word w_i and v_j, *e.g.*, cosine, dot product [2]
- **Feature Extraction and Regression**: $\Phi(\cdot)$: word embedding. $\mathrm{F}(\cdot)$: scoring function [2]

Credits: [1] Sun *et al.*, Sentence Similarity Based on Contexts, In TACL'22. [2] Pang *et al.*, Text Matching as Image Recognition, In AAAI'16.

Part 4 — Matrix-based Methods

- **Key Idea**: construct a **similarity matrix** between two sentences, each element of which represents the similarity between the two corresponding units in two sentences. Then, the matrix is aggregated in different ways to induce the **final similarity score** [1]

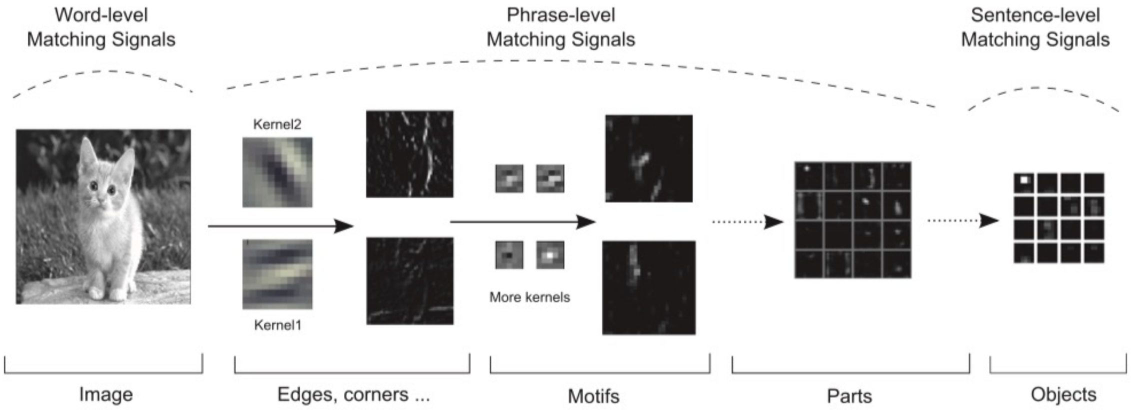


Figure 2: Relationships between text matching and image recognition.

Credits: [1] Sun *et al.*, Sentence Similarity Based on Contexts, In TACL'22. [2] Pang *et al.*, Text Matching as Image Recognition, In AAAI'16.

Part 4 — Matrix-based Methods

- **Similarity Matrix**:
  - Word2Vec → Cosine and dot product [1]
  - Bi-LSTMs → Pairwise interaction (Cosine, L_2, dot product) → Similarity focus via Weight [2]
  - MultiGranCNN → radial basis function kernel, inner product and sigmoid, weighted concat [3]

- **Feature Extraction and Regression**:
  - 2-layer CNN + 1-layer MLP [1]
  - 19-layer Deep ConvNet + 2-layer MLP [2]
  - mfCNN + 2-layer MLP [3]

Credits:
[1] Pang *et al.*, Text Matching as Image Recognition, In AAAI'16.
[2] He *et al.*, Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement, In NAACL'16.
[3] Yin *et al.*, MultiGranCNN: An Architecture for General Matching of Text Chunks on Multiple Levels of Granularity, In ACL'15.

Part 5 — Alignment/Attention-based Methods

- **Key Idea**: Aligner aligns related words in two sentences

  Semantic similarity is a monotonically increasing function of the degree to which [1, 2]

  - The two sentences **contain similar semantic units**
  - Such units **occur in similar semantic contexts** in the respective sentences

- **Output**: Predict the pair's semantic similarity by **taking the proportion of their aligned content words**
- **Intuition**: More aligned semantic components ⇒ higher semantic similarity

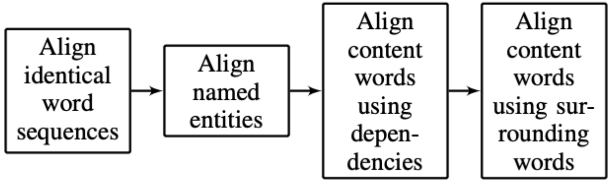


Figure 1: The alignment pipeline.

Credits:
[1] Sultan *et al.*, Back to Basics for Monolingual Alignment: Exploiting Word Similarity and Contextual Evidence, In TACL'14.
[2] Sultan *et al.*, DLS@CU: Sentence Similarity from Word Alignment, In SemEval'14.

Part 5 — Alignment/Attention-based Methods

- **Key Idea**: semantic alignment and comparison of two text sequences via Attention Mechanism
- **How**: Inter-sequence alignment layer

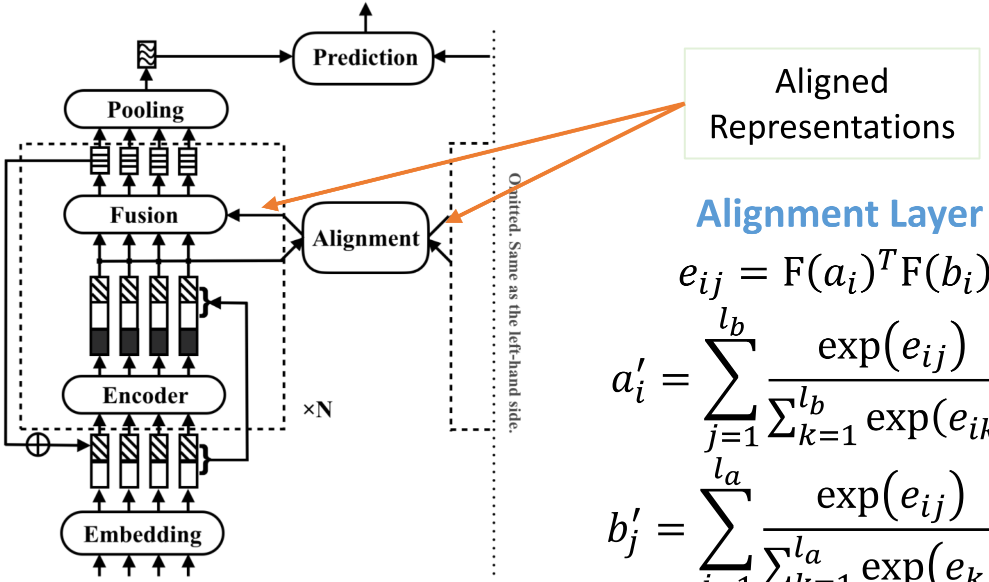
$$\hat{\mathrm{y}} = H([\mathrm{v}_1;\ \mathrm{v}_2; |\mathrm{v}_1 - \mathrm{v}_2|; \mathrm{v}_1 \circ \mathrm{v}_2]).$$

Fusion Layer

$$\bar{a}_i^1 = G_1([a_i; a_i']),$$
$$\bar{a}_i^2 = G_2([a_i; a_i - a_i']),$$
$$\bar{a}_i^3 = G_3([a_i; a_i \circ a_i']),$$
$$\bar{a}_i = G([\bar{a}_i^1;\ \bar{a}_i^2;\ \bar{a}_i^3]).$$

Augmented Residual Connections

$$x_i^{(n)} = \left[x_i^{(1)};\ o_i^{(n-1)} + o_i^{(n-2)}\right].$$



Alignment Layer

$$e_{ij} = \mathrm{F}(a_i)^T \mathrm{F}(b_i),$$

$$a_i' = \sum_{j=1}^{l_b} \frac{\exp(e_{ij})}{\sum_{k=1}^{l_b} \exp(e_{ik})} b_j,$$

$$b_j' = \sum_{i=1}^{l_a} \frac{\exp(e_{ij})}{\sum_{k=1}^{l_a} \exp(e_{kj})} a_i.$$

Credits: Yang *et al.*, Simple and Effective Text Matching with Richer Alignment Features, In ACL'19.

Part 6 – Word Distance-based Methods

- **Key Idea**: calculate the **cost of transferring** from one sentence to another
- **Intuition**: smaller cost $\Rightarrow$ more similar sentences
- **How**: **Earth Mover's Distance** (optimal transport cost) – the minimum cost required to turn one pile of dirt into another pile of dirt [1]
- **How**: **Word Mover's Distance** (WMD) – measure the dissimilarity between two documents as the **minimum amount of distance** that the embedded words of one document need to transform to words of another document [2]
- Each point x_i has a **probability mass** $m_i \in [0, 1]$. $c(\cdot)$ is a **transportation cost function**

$$\mu = \{(x_i, m_i)\}_{i=1}^{n}, \quad \mu' = \{(x'_j, m'_j)\}_{j=1}^{n'}.$$

$c(x_i, x'_j)$: determines the transportation cost per unit amount (distance) between two points x_i and x'_j.

Credits: [1] Villani *et al.*, Optimal Transport, In Springer'09. [2] Kusner *et al.*, From Word Embeddings to Document Distances. In ICML'15.

Part 6 — Word Distance-based Methods

- **Key Idea**: calculate the cost of transferring from one sentence to another
- **How**: Earth Mover's Distance (optimal transport cost) — the minimum cost required to turn one pile of dirt into another pile of dirt
- Each point x_i has a probability mass $m_i \in [0,1]$. $c(\cdot)$ is a transportation cost function

$$\mu = \{(x_i, m_i)\}_{i=1}^{n}, \quad \mu' = \{(x'_j, m'_j)\}_{j=1}^{n'}.$$

$c(x_i, x'_j)$: determines the transportation cost per unit amount (distance) between two points x_i and x'_j $\Longrightarrow$ Alignment if c is small

$$\mathrm{EMD}(\mu, \mu'; c) := \min_{T \in \mathbb{R}_{\geq 0}^{n \times n'}} \sum_{i,j} T_{ij} c(x_i, x'_j),$$

$$\text{s.t.} \begin{cases} T\mathbb{1}_n = \mathrm{m} := (m_1, \ldots, m_n)^T, \\ T^T\mathbb{1}_{n'} = \mathrm{m}' := (m'_1, \ldots, m'_{n'})^T. \end{cases}$$

$\mathrm{EMD}(\mu, \mu'; c)$ is the cost of the best transportation plan between two distributions μ and μ'

$T \in \mathbb{R}_{\geq 0}^{n \times n'}$ denotes a transportation plan, where each element T_{ij} represents the mass transported from x_i to x'_j.

Credits: Yokoi *et al.*, Word Rotator's Distance, In EMNLP'20.

Part 6 — Word Distance-based Methods

- **Key Idea**: calculate the cost of transferring from one sentence to another
- **How**: Word Mover's Distance (WMD): is the cost of transporting a set of word vectors in an embedding space (Euclidean space)
- Each sentence s as a uniformly weighted distribution μ_s comprising word vectors w_i
- Transportation cost between word vectors $c_{\mathrm{E}}(w_i, w_j')$ is represented by Euclidean distance

$$\mu_s := \left\{\left(w_i, \frac{1}{n}\right)\right\}_{i=1}^{n},$$

$$\mu_s' := \left\{\left(w_j', \frac{1}{n'}\right)\right\}_{j=1}^{n'},$$

$$c_{\mathrm{E}}(w_i, w_j') := \|w_i - w_j'\|,$$

$$\mathrm{WMD}(s, s') = \mathrm{EMD}(\mu, \mu'; c_{\mathrm{E}}).$$

$\mathrm{WMD}(s, s')$ is defined as the EMD between two such distributions using the cost function $c_{\mathrm{E}}(w_i, w_j')$

Credits: Yokoi *et al.*, Word Rotator's Distance, In EMNLP'20.

Part 7 — Sentence Embedding-based Methods

Word → Sentence?

Linguistics = $\begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$

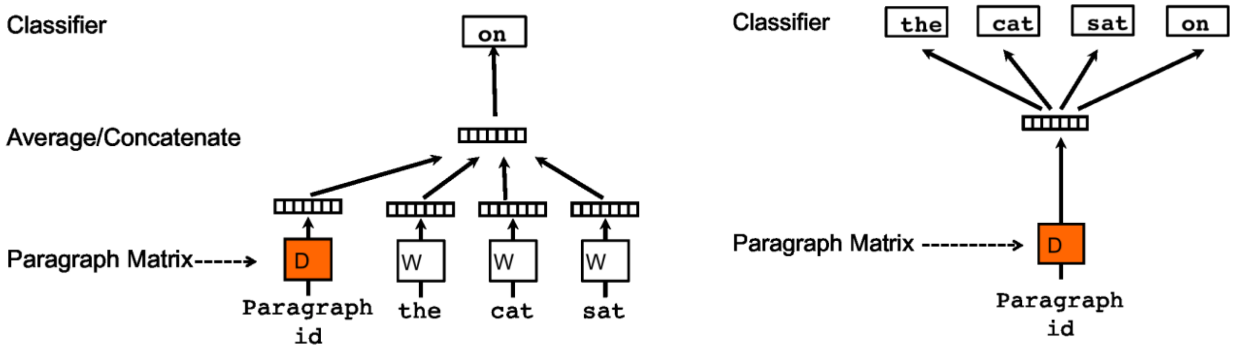
Natural language processing is fun. = $\begin{pmatrix} -0.132 \\ 1.129 \\ 0.827 \\ 0.110 \\ -0.527 \\ 0.156 \\ 0.349 \\ -0.286 \end{pmatrix}$

- Compute sentence similarity, *e.g.*, using the inner product
- Use as features for downstream tasks, *e.g.*, sentence classification

Credits: [CS224N Research Highlight (Presenter: Danqi Chen@Princeton)]

Part 7 – Sentence Embedding-based Methods

- **Key Idea**: **high-dimensional representations for sentences**. They are expected to contain rich sentence semantics so that the similarity between two sentences can be computed by **considering their sentence embeddings** via certain metrics such as cosine similarity
- **Paragraph Vector**: **fixed-length feature representations** from variable-length pieces of texts, such as sentences, paragraphs, and documents



Credits: Le *et al.*, Distributed Representations of Sentences and Documents, In ICML'14.

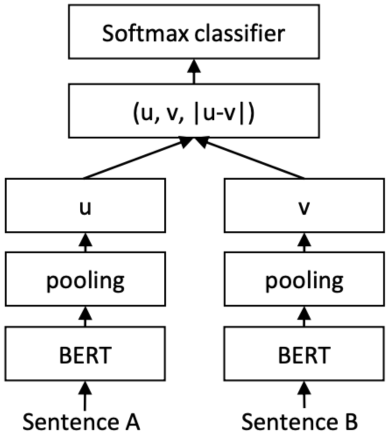
Part 7 — Sentence Embedding-based Methods

- **Paragraph Vector**: **fixed-length feature representations** from variable-length pieces of texts [1]
- Skip-Thought Vectors [2]
- Smooth Inverse Frequency (SIF) [3]
- Sequential Denoising Autoencoders (SDAEs) [4]
- InferSent [5]
- Quick-Thought Vectors [6]
- Universal Sentence Encoder [7]

Credits:
[1] Le *et al.*, Distributed Representations of Sentences and Documents, In ICML'14.
[2] Kiros *et al.*, Skip-Thought Vectors, In NeurIPS'15.
[3] Arora *et al.*, A Simple but Tough-to-beat Baseline for Sentence Embeddings, In ICLR'17.
[4] Hill *et al.*, Learning Distributed Representations of Sentences from Unlabelled Data, In NAACL'16.
[5] Conneau *et al.*, Supervised Learning of Universal Sentence Representations from Natural Language Inference Data, In EMNLP'17.
[6] Logeswaran *et al.*, An Efficient Framework for Learning Sentence Representations, In ICLR'18.
[7] Cer *et al.*, Universal Sentence Encoder, In arXiv'18.

Part 7 – Sentence Embedding-based Methods

- **Sentence Embedding**: **distributed representation of a sentence** in the form of a vector which encodes meaningful semantic information [Wikipedia]
- **Key Idea**: produce sentence embeddings based on the **pretraining-finetuning paradigm using large-scale unlabeled corpora**
- Sentence-BERT (SBERT) [1]
- SBERT-WK [2]
- BERT-flow [3]
- BERT-whitening [4]



Credits:
[1] Reimers *et al.*, Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, In EMNLP’19.
[2] Wang *et al.*, SBERT-WK: A Sentence Embedding Method by Dissecting BERT-based Word Models, In IEEE/ACM T-ASLP’20.
[3] Li *et al.*, On the Sentence Embeddings from Pre-trained Language Models, In EMNLP’20.
[4] Su *et al.*, Whitening Sentence Representations for Better Semantics and Faster Retrieval, In arXiv’21.

Part 7 — Sentence Embedding-based Methods

- **BERT-based Scores Key Idea**: automatic evaluation metric for text generation

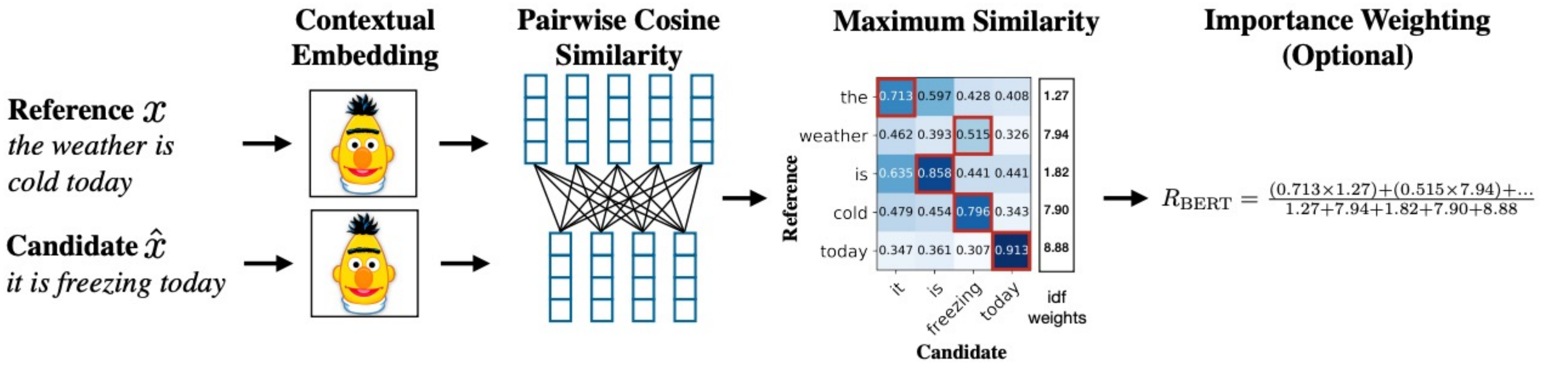


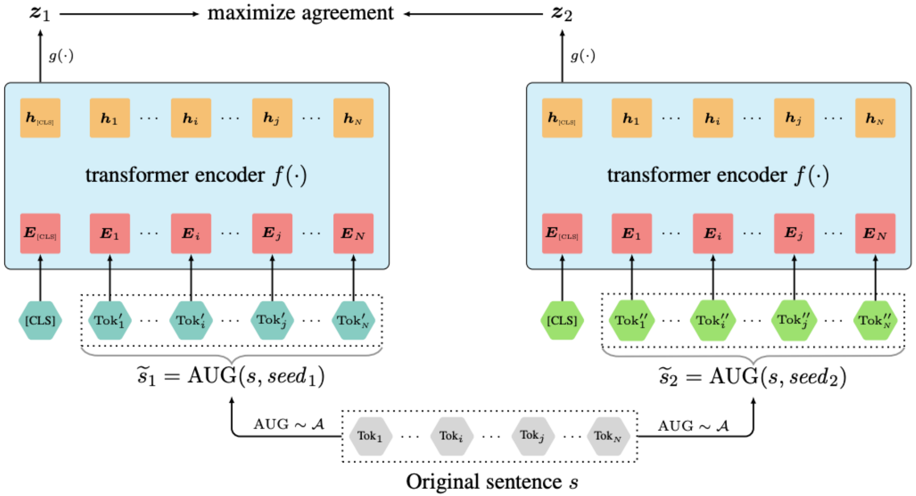
Figure 1: Illustration of the computation of the recall metric R_{BERT}. Given the reference x and candidate $\hat{x}$, we compute BERT embeddings and pairwise cosine similarity. We highlight the greedy matching in red, and include the optional idf importance weighting.

Credits:

[1] Zhang *et al.*, BERTScore: Evaluating Text Generation with BERT, In ICLR’20.
[2] Sellam *et al.*, BLEURT: Learning Robust Metrics for Text Generation, In ACL’20.

Part 7 — Sentence Embedding-based Methods

- **Contrastive Learning Key Idea**: two similar sentences are pulled close, and two random sentences are pulled away in the sentence representation space



Credits:

[1] Wu *et al.*, CLEAR: Contrastive Learning for Sentence Representation, In arXiv'20.
[2] Carlsson *et al.*, Semantic Re-tuning with Contrastive Tension, In ICLR'21.
[3] Kim *et al.*, Self-Guided Contrastive Learning for BERT Sentence Representations, In ACL'21.
[4] Yan *et al.*, ConSERT: A Contrastive Framework for Self-Supervised Sentence Representation Transfer, In ACL'21.
[5] Gao *et al.*, SimCSE: Simple Contrastive Learning of Sentence Embeddings, In EMNLP'21.

Part 8 — Evaluation Metrics for *Semantic Textual Similarity*

- **Pearson correlation** (Pearson Linear Correlation Coefficient) r — **measure the prediction accuracy**

$$r = \frac{\sum_{i=1}^{n}(s_i - \bar{s})\,(q_i - \bar{q})}{\sqrt{\sum_{i=1}^{n}(s_i - \bar{s})^2}\,\sqrt{\sum_{i=1}^{n}(q_i - \bar{q})^2}},$$

where s_i and q_i are the gold label and the model's prediction of the i-th sentence. $\bar{s}$ and $\bar{q}$ are the mean values of $\mathbf{s}$ and $\mathbf{q}$. n is the number of sentences.

- **Spearman's rank correlation** (Spearman's Rank-order Correlation Coefficient) ρ — **measure the prediction monotonicity**

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2 - 1)},$$

where d_i is the difference between the i-th sentence's rank in the model's predictions and gold labels.

Part 9 — Benchmark Databases

0: Dissimilar

Synonymous

Dataset Name	Word/Sentence pairs	Similarity score range	Year	Reference
R&G	65	0–4	1965	[107]
M&C	30	0–4	1991	[78]
WS353 Relatedness	353	0–10	2002	[30]
LiSent	65 Sentences	0–4	2007	[63]
SRS Biomedical	30	0–4	2007	[94]
WS353-Sim	203	0–10	2009	[1]
STS2012	5,250 Sentences	0–5	2012	[5]
STS2013	2,250 Sentences	0–5	2013	[6]
WP300	300	0–1	2013	[61]
STS2014	3,750 Sentences	0–5	2014	[3]
SL7576	7,576	1–5	2014	[116]
SimLex-999	999	0–10	2014	[40] 900 Similar Word Paris
SICK-Relatedness	10,000 Sentences	1–5	2014	[69]
STS2015	3,000 Sentences	0–5	2015	[2]
SimVerb	3,500	0–10	2016	[34]
STS2016	1,186 Sentences	0–5	2016	[4]
WiC	5,428	NA	2019	[97]
STS Benchmark	8628	0-5	2017	[Link]

Credits: Chandrasekaran *et al.*, Evolution of Semantic Similarity—A Survey, In ACM Computing Survey'21.

Part 10 – Takeaways

- **Distance Measurement**
- **Spatial Metrics**
  - Cosine Similarity
  - Manhattan Distance or L_1 Norm
  - Euclidean Distance or L_2 Norm
- **Mutual Information**
  - Hindle
  - Lin
- **Relative Entropy**
  - Kullback-Leibler Divergence (KLD) – Common Occurrence, Absolute, Average, and Maximum
  - $\boldsymbol{\alpha}$-skew Divergence (ASD)
  - Jensen-Shannon Divergence (JSD)
- **Others**
  - Co-occurrence Retrieval Models (CRM), Dice Coefficient, Division Measure, Jaccard, Product Measure

Credits: Mohammad *et al.*, Distributional Measures of Semantic Distance: A Survey, In arXiv'12.

Part 10 – Takeaways

Evolution of Semantic Similarity

- **Knowledge-based Methods**

  Pros: consider the actual meaning of the text

  Cons: not adaptable across different domains and languages

- **Corpus-based Methods**

  Pros: have a statistical background and can be implemented across languages

  Cons: don’t consider the actual meaning of the text

- **Deep Learning-based Methods**

  Pros: better performance

  Cons: require high computational resources and lack interpretability

- **Hybrid Methods**

  Pros: take advantage of the benefits of different methods

Credits: Chandrasekaran *et al.*, Evolution of Semantic Similarity—A Survey, In ACM Computing Survey’21.

Part 10 – Takeaways (This one is generated by ChatGPT)

➢ **Deep Learning-based Methods** (Note: Some works may combine the following ideas.)

- **Matrix-based Methods**

  Pros: scalability for handling large datasets efficiently and Interpretability for clear alignment between words

  Cons: sensitivity to length (larger matrices and potentially skewed similarity scores) and sparse matrices that make it challenging to capture nuanced relationships between words.

- **Alignment-based Methods**

  Pros: better identify word-level and fine-grained correspondences between sentences

  Cons: computationally intensive and not robust to paraphrasing

- **Word Distance-based Methods**

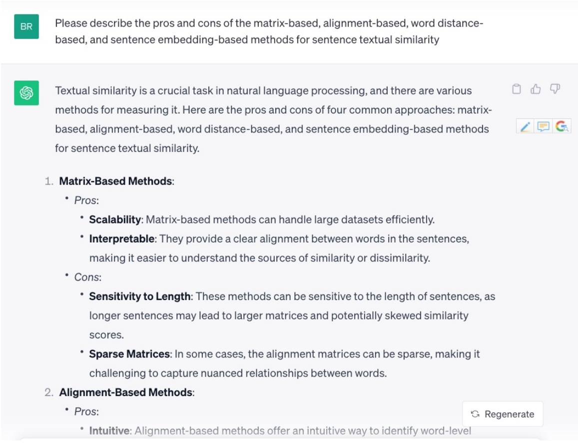
  Pros: straightforward to implement and sensitive to Word Choice

  Cons: order insensitivity and lack of context

- **Sentence Embedding-based Methods**

  Pros: robust to syntax and versatile

  Cons: loss of fine-grained Information and embedding quality



Part 10 — Takeaways

- **Evaluation Metrics**:
  - **Pearson correlation** r — **measure the prediction accuracy**

$$r = \frac{\sum_{i=1}^{n}(s_i - \bar{s})(q_i - \bar{q})}{\sqrt{\sum_{i=1}^{n}(s_i - \bar{s})^2}\sqrt{\sum_{i=1}^{n}(q_i - \bar{q})^2}}.$$

  - **Spearman's rank correlation** ρ — **measure the prediction monotonicity**

$$\rho = 1 - \frac{6\sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}.$$

- **Benchmark Databases**:
  - STS12, STS13, STS14, STS15, STS16, STS Benchmark (STSb), and SICK-Relatedness
- **Note**: Other related tasks have their own set of benchmarks and specific evaluation metrics.

Thank you very much for your attention!

Dependable Computing Laboratory,
Department of Electrical and Computer Engineering,
Boston University

BOSTON UNIVERSITY