Topics in Natural Language Processing

Nils Kujath

v2026.05

Topic 1

Word Embeddings

Degenerate Geometry of One-Hot Word Representations

- Let $V = \{w_1, \ldots, w_{|V|}\}$ be a finite vocabulary. The canonical (standard) basis of $\mathbb{R}^{|V|}$ is the indexed set $\mathcal{B}_{\text{can}} = \{\mathbf{e}_1, \ldots, \mathbf{e}_{|V|}\}$, where $\mathbf{e}_i$ denotes the i-th basis vector, i.e. the unique (one-hot) vector with a 1 in coordinate i and zeros elsewhere. The BoW construction $\boxed{\iota : V \to \mathbb{R}^{|V|}, \quad w_i \mapsto \mathbf{e}_i}$ identifies each word $w_i \in V$ with the canonical basis vector $\mathbf{e}_i \in \mathbb{R}^{|V|}$.
- Under the standard inner product on $\mathbb{R}^{|V|}$, all distinct word types are mutually orthogonal:

$$\boxed{\mathbf{e}_u{}^{\top}\mathbf{e}_v = \delta_{uv} = \begin{cases} 1 & \text{if } u = v, \\ 0 & \text{if } u \neq v \end{cases}, \quad \forall u, v \in \{1, \ldots, |V|\}}.$$

  Fmr., the metric restricted to $\iota(V)$ is trivial; all word types are equidistant in $\mathbb{R}^{|V|}$: $\forall u, v \in \{1, \ldots, |V|\}$,

$$\boxed{d\big|_{\iota(V)\times\iota(V)}(\mathbf{e}_u, \mathbf{e}_v) = \|\mathbf{e}_u - \mathbf{e}_v\| = \begin{cases} 0 & \text{if } u = v, \\ \sqrt{\underbrace{(1-0)^2}_{\text{pos. } u} + \underbrace{(0-1)^2}_{\text{pos. } v} + \underbrace{(0-0)^2 + \cdots + (0-0)^2}_{|V|-2 \text{ pos.}}} = \sqrt{2} & \text{if } u \neq v \end{cases}}.$$

- Consequently, the geometry induced by ι is degenerate: all distinct words are equally dissimilar, and no notion of graded semantic proximity can be expressed under the BoW representation.

From the Distributional Hypothesis to Word Embeddings

- Let $V = \{w_1, \ldots, w_{|V|}\}$ be a finite vocabulary. Let $\mathcal{D} = (t_1, t_2, \ldots, t_N)$ be a corpus of N tokens from V. Fix a context window size $k \in \mathbb{N}^+$. Define the context map $\mathcal{C}_k : \{n \in \mathbb{N} : k < n \leq N - k\} \to V^{2k}$ by:

$$\boxed{\mathcal{C}_k(n) = (t_{n-k}, \ldots, t_{n-1}, t_{n+1}, \ldots, t_{n+k})}.$$

  Note that positions $n \leq k$ and $n > N - k$ are excluded since k tokens of context are required on each side.

- For each $w_i \in V$, define the distributional profile of w_i in $\mathcal{D}$ as the multiset:

$$\boxed{\Delta_k(w_i) = \{\!\{\mathcal{C}_k(n) : n \in \{k{+}1, \ldots, N{-}k\},\ t_n = w_i\}\!\}}.$$

- The Distributional Hypothesis (see esp. Harris 1954 and Firth 1957) asserts that w_i and w_j are semantically similar if they appear in a similar contexts, that is, if $\Delta_k(w_i) \approx \Delta_k(w_j)$. The previous slide has shown that the degenerate geometry of BoW representations precludes any graded notion of similarity. However, comparing multisets over V^{2k} directly also seems intractable. The goal is therefore to find a map:

$$\boxed{\phi : V \to \mathbb{R}^m \, (m \ll |V|) \quad \text{s.t.} \quad \Delta_k(w_i) \approx \Delta_k(w_j) \quad \text{is operationalised as} \quad \phi(w_i) \approx \phi(w_j) \text{ in } \mathbb{R}^m}.$$

  That is, ϕ embeds the discrete set V into (the so-called embedding space) $\mathbb{R}^m$ such that distributional similarity in $\mathcal{D}$ is faithfully compressed into geometric proximity. (Note: We will discuss later why we desire $m \ll |V|$.)

Count-Based Word Embeddings (Schütze 1992)

- Let $V = \{w_1, \ldots, w_{|V|}\}$ be a finite vocabulary and $k \in \mathbb{N}^+$ the selected size of the context window. We could define a co-occurrence matrix $M \in \mathbb{N}^{|V| \times |V|}$ (see Schütze 1992 for this idea) where $M_{[i,j]}$ is the number of times w_j appears in a context window of size k around w_i in the corpus $\mathcal{D} = (t_1, \ldots, t_N)$:

$$M_{[i,j]} = \sum_{n=k+1}^{N-k} \underbrace{\mathbf{1}[t_n = w_i]}_{1 \text{ if center is } w_i} \cdot \sum_{\substack{l=n-k \\ l \neq n}}^{n+k} \underbrace{\mathbf{1}[t_l = w_j]}_{1 \text{ if context slot is } w_j}.$$

  The outer sum ranges over all valid center positions $n \in \{k+1, \ldots, N-k\}$; the inner sum scans the $2k$ surrounding context slots.

- Recall the context map $\mathcal{C}_k(n) = (t_{n-k}, \ldots, t_{n-1}, t_{n+1}, \ldots, t_{n+k})$ for $n \in \{k+1, \ldots, N-k\}$, and the distributional profile $\Delta_k(w_i) = \{\!\{\mathcal{C}_k(n) : n \in \{k+1, \ldots, N-k\},\ t_n = w_i\}\!\}$ from the previous slide. The co-occurrence matrix M is a lossy compression of the distributional profiles Δ_k over $\mathcal{D}$: ordering within each context tuple is discarded, and only co-occurrence frequencies are retained.
- In M, each row $\mathbf{m}_i = (M_{[i,1]}, \ldots, M_{[i,|V|]}) \in \mathbb{R}^{|V|}$ is already a representation of w_i that reflects distributional similarity: words with similar co-occurrence patterns have similar row vectors. However, these rows live in $\mathbb{R}^{|V|}$, not the $\mathbb{R}^m$ with $m \ll |V|$ sought on the previous slide.

PPMI Reweighting of Count-based Co-Occurrence Matrices (Bullinaria & Levy 2007)

- The entries of M suffer from frequency dominance (Zipf 1935; Luhn 1958; Spärck Jones 1972). A remedy is Pointwise Mutual Information (PMI; Fano 1961), originally applied to lexical co-occurrence data by Church & Hanks (1990) and later used to reweight co-occurrence matrices by Bullinaria & Levy (2007). PMI replaces each raw count $M_{[i,j]}$ with a score that factors out the frequency effect, yielding $M^{\mathrm{PMI}} \in \mathbb{R}^{|V|\times|V|}$:

$$M^{\mathrm{PMI}}_{[i,j]} = \mathrm{PMI}(w_i, w_j) = \underbrace{\log_2}_{\substack{\text{maps to symmetric}\\ \text{scale centred at } 0}} \frac{\underbrace{P_{\mathcal{D}}(w_i, w_j)}_{\text{observed co-occurrence}}}{\underbrace{P_{\mathcal{D}}(w_i)\cdot P_{\mathcal{D}}(w_j)}_{\text{chance-level co-occurrence}}} = \log_2 \frac{\dfrac{M_{[i,j]}}{\sum_{a=1}^{|V|}\sum_{b=1}^{|V|} M_{[a,b]}}}{\dfrac{\mathrm{count}(w_i,\mathcal{D})}{|\mathcal{D}|}\cdot\dfrac{\mathrm{count}(w_j,\mathcal{D})}{|\mathcal{D}|}}$$

- In practice, most word pairs never co-occur at all ($M_{[i,j]} = 0$, sending $\mathrm{PMI} \to -\infty$), and pairs with very low counts produce large negative values that reflect data sparsity rather than genuine anti-association. The standard solution is to clamp all negative values to zero, yielding Positive PMI (PPMI; see Bullinaria & Levy 2007):

$$M^{\mathrm{PPMI}}_{[i,j]} = \mathrm{PPMI}(w_i, w_j) = \max\big(0,\ \mathrm{PMI}(w_i, w_j)\big).$$

  A row $M^{\mathrm{PPMI}}_{[i,*]} \in \mathbb{R}^{1\times|V|}$ cast as a vector in $\mathbb{R}^{|V|}$ could now serve as a word vector for $w_i \in V$. Though this solves the frequency problem, the resulting embeddings still do not live in the desired space $\mathbb{R}^m$ where $m \ll |V|$.

Dimensionality Reduction via Truncated Singular Value Decomposition

- The matrix $M^{\mathrm{PPMI}} \in \mathbb{R}^{|V| \times |V|}$ from the previous slide yields word vectors in $\mathbb{R}^{|V|}$. To obtain vectors in the desired $\mathbb{R}^m$ where $m \ll |V|$, we apply the Singular Value Decomposition (SVD), following a line of work that applied SVD to term-document matrices (Deerwester et al. 1990), then to count-based co-occurrence matrices (Schütze 1992), and finally to PPMI-reweighted co-occurrence matrices (Bullinaria & Levy 2012).
- SVD decomposes M^{PPMI} into a set of orthogonal axes, each associated with a singular value σ_i that measures how much of the matrix's structure that axis captures. These axes are sorted by importance: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{|V|} \geq 0$. The truncated SVD of rank m retains only the m axes with the largest singular values and discards the rest (Eckart & Young 1936):

$$M^{\mathrm{PPMI}} \approx \underbrace{U_m}_{\in \mathbb{R}^{|V| \times m}} \quad \underbrace{\Sigma_m}_{\in \mathbb{R}^{m \times m}} \quad \underbrace{V_m^\top}_{\in \mathbb{R}^{m \times |V|}}$$

- The full product $U_m \Sigma_m V_m^\top$ would reconstruct a $|V| \times |V|$ matrix. The compression comes from stopping before the last multiplication: row i of $U_m \Sigma_m \in \mathbb{R}^{|V| \times m}$ is a word vector for w_i in $\mathbb{R}^m$. The m dimensions no longer correspond to individual context words as in M^{PPMI}; they are abstract axes that capture the most important co-occurrence patterns across the entire vocabulary. This finally delivers the embedding $\phi : V \to \mathbb{R}^m$ with $m \ll |V|$. An alternative approach is to learn word vectors in $\mathbb{R}^m$ directly from $\mathcal{D}$ (see the following slides).

Excursus 1

Fully-Connected Feed-Forward Neural Networks (FFNNs)

Architecture of Fully-Connected Feed-Forward Neural Networks (1)

- The following architectural choices must be made before training the FFNN (this is usually referred to as the setting of the network's hyperparameters):
  - The number of layers $L \in \mathbb{N}_{\geq 1}$, corresponding to the number of hidden layers plus the output layer;
  - The width of each layer, written as $(d_0, d_1, \ldots, d_L)$, where d_0 is the dimension of the input vector, d_ℓ for $1 \leq \ell \leq L-1$ the number of neurons in hidden layer ℓ, and d_L the dimension of the output layer; and
  - The activation functions $g^{(\ell)} : \mathbb{R}^{d_\ell} \to \mathbb{R}^{d_\ell}$ for each layer $1 \leq \ell \leq L$, written as $(g^{(1)}, \ldots, g^{(L)})$.
- The (trainable) parameters of the network are collected in a set $\theta = \{(\mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)})\}_{\ell=1}^{L}$, where for each layer $\ell = 1, \ldots, L$, $\mathbf{W}^{(\ell)} \in \mathbb{R}^{d_{\ell-1} \times d_\ell}$ is the weight matrix and $\mathbf{b}^{(\ell)} \in \mathbb{R}^{1 \times d_\ell}$ is the bias row vector.
- The neural network itself can be seen as the composition of layer functions: $F_\theta = f^{(L)} \circ f^{(L-1)} \circ \cdots \circ f^{(1)}$, with $F_\theta : \mathbb{R}^{1 \times d_0} \to \mathbb{R}^{1 \times d_L}$, where each layer function is defined as:

$$\boxed{f^{(\ell)} : \mathbb{R}^{1 \times d_{\ell-1}} \to \mathbb{R}^{1 \times d_\ell}, \qquad \mathbf{u} \mapsto g^{(\ell)}\left(A(\mathbf{u})\, \tilde{\mathbf{W}}^{(\ell)}\right)}$$

  Here $\mathbf{u} \in \mathbb{R}^{1 \times d_{\ell-1}}$ is the input to layer ℓ, $A(\mathbf{u}) = (1, \mathbf{u}) \in \mathbb{R}^{1 \times (d_{\ell-1}+1)}$ is the augmentation operator, and $\tilde{\mathbf{W}}^{(\ell)}$ is the augmented weight matrix that integrates the bias into the weight matrix, so that the affine map $\mathbf{u}\mathbf{W}^{(\ell)} + \mathbf{b}^{(\ell)}$ can be written as the linear map $A(\mathbf{u})\, \tilde{\mathbf{W}}^{(\ell)}$: $\tilde{\mathbf{W}}^{(\ell)\top} = \begin{bmatrix} \mathbf{b}^{(\ell)\top} & \mathbf{W}^{(\ell)\top} \end{bmatrix} \in \mathbb{R}^{d_\ell \times (d_{\ell-1}+1)}$.

Architecture of Fully-Connected Feed-Forward Neural Networks (2)

- Given $\mathbf{x} \in \mathbb{R}^{1\times d_0}$, forward propagation evaluates F_θ by computing activations $\mathbf{a}^{(0)} := \mathbf{x}$ and, for $\ell = 1, \ldots, L$:

$$\boxed{\tilde{\mathbf{a}}^{(\ell-1)} := A(\mathbf{a}^{(\ell-1)}) \in \mathbb{R}^{1\times(d_{\ell-1}+1)}, \quad \mathbf{z}^{(\ell)} := \tilde{\mathbf{a}}^{(\ell-1)}\tilde{\mathbf{W}}^{(\ell)} \in \mathbb{R}^{1\times d_\ell}, \quad \mathbf{a}^{(\ell)} := g^{(\ell)}(\mathbf{z}^{(\ell)}) \in \mathbb{R}^{1\times d_\ell}}.$$

  The output is the final activation: $F_\theta(\mathbf{x}) := \mathbf{a}^{(L)}$. In classification with $\mathcal{Y} = \{1, \ldots, C\}$ and $d_L = C$, a prediction is obtained via $\hat{y}(\mathbf{x}) := \arg\max_{c\in\mathcal{Y}} a_c^{(L)}$, or as a distribution: $P(y = c \mid \mathbf{x}) := \mathrm{softmax}(\mathbf{a}^{(L)})_c$. In regression, typically $d_L = 1$ with identity activation, so $F_\theta(\mathbf{x}) = a_1^{(L)} \in \mathbb{R}$.
- If all activations are linear (or affine), the composition of layers reduces to a single affine map; adding depth does not increase expressive power. Such a network can only represent linear decision boundaries (hyperplanes) and fails on datasets that are not linearly separable (see Minsky & Papert 1969).
  A dataset $X = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^T \subset \mathbb{R}^d \times \{1, \ldots, C\}$ is linearly separable if $\exists\, (\mathbf{W}, \mathbf{b}) \in \mathbb{R}^{d\times C} \times \mathbb{R}^C$ s.t.:

$$\boxed{\forall i \in \{1, \ldots, T\},\ (\forall c \in \{1, \ldots, C\} \setminus \{y^{(i)}\}) : (\mathbf{x}^{(i)}\mathbf{W}_{[*,\, y^{(i)}]} + b_{y^{(i)}}) > (\mathbf{x}^{(i)}\mathbf{W}_{[*,\, c]} + b_c)}.$$

- Universal Approximation Theorem: FFNNs with at least one hidden layer and a non-constant, non-linear activation (e.g. ReLU, sigmoid) are universal approximators of continuous functions on compact subsets of $\mathbb{R}^D$ (see Cybenko 1989 and Hornik et al. 1989; see also Leshno et al. 1993), that are expressive enough to perfectly separate any finite dataset, though this does not imply such solutions can be efficiently learned or will generalise.

Learning as Minimizing the Loss Function

- Given a training set $X = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{T}$ and a scoring function F_θ, a loss function $L(X, \theta)$ measures how poorly the system performs on X, typically by aggregating a per-example loss $l(\mathbf{x}, y, \theta)$. Learning means finding the parameters that minimize this loss: $\boxed{\theta^* = \arg\min_\theta L(X, \theta) = \arg\min_\theta \frac{1}{T} \sum_{i=1}^{T} l(\mathbf{x}^{(i)}, y^{(i)}, \theta)}$.
- Multi-class classification: Let $\mathcal{Y} = \{1, \ldots, C\}$ be the set of class labels and $F_\theta(\mathbf{x}) \in \mathbb{R}^{1 \times C}$ be the output of the scoring function. The per-example margin $\mathrm{margin}(\mathbf{x}, y, F_\theta) = F_\theta(\mathbf{x})_y - \max_{c \neq y} F_\theta(\mathbf{x})_c$ quantifies how confidently an example is classified. The hinge loss penalizes training examples whose margin is smaller than a fixed threshold (usually 1). For an ex. $(\mathbf{x}, y)$, it is def. as $\boxed{l_{\text{hinge}}(\mathbf{x}, y, \theta) = \max\big(0,\ 1 - \mathrm{margin}(\mathbf{x}, y, F_\theta)\big)}$ and is equal to zero when the margin is sufficiently large, while increasing linearly for examples that are misclassified or classified with insufficient confidence. (The log loss can be seen as a smooth approx. of the hinge loss: it penalizes small margins continuously, equals 1 if the margin is 0, and smoothly decreases towards 0 as the margin grows.)
- Probabilistic classification: the standard loss is the cross-entropy loss. Given a model defining conditional class probabilities $P(y \mid \mathbf{x}, \theta)$, learning consists in maximizing the conditional likelihood of the observed labels, which is equivalent to minimizing the negative log likelihood of the observed labels: $\boxed{l_{\text{CE}}(\mathbf{x}, y, \theta) = -\log P(y \mid \mathbf{x}, \theta)}$.
- Regression: Let $F_\theta(\mathbf{x}) \in \mathbb{R}$ be the output of the scoring function. The standard loss is the mean squared error (MSE), minimized ($= 0$) when predicted values exactly match target values: $\boxed{l_{\text{MSE}}(\mathbf{x}, y, \theta) = \big(F_\theta(\mathbf{x}) - y\big)^2}$.

Learning the Parameters: Full-Batch, Stochastic, and Mini-Batch Gradient Descent

- Learning means minimizing $J(X, \theta)$ (typically a loss function + a regularization term), i.e., we need to find how to update θ such that J decreases: $\boxed{\theta^\star = \arg\min_\theta J(X, \theta)}$. Gradient descent is an iterative optimization algo. that updates the parameters in the direction of steepest decrease of J: $\boxed{\theta^{(t+1)} = \theta^{(t)} - \eta \nabla_\theta J\left(X, \theta^{(t)}\right)}$ where $\nabla_\theta J$ is the gradient of J with respect to the parameters, and $\eta > 0$ is the learning rate (step size).
- For Full-Batch Gradient Descent, at each iteration, we perform a forward pass, compute the loss $J(X, \theta)$ on the full dataset, compute its gradient via backpropagation, and update all parameters θ. If J is convex and η is sufficiently small, gradient descent converges to the global minimum.
- J usually decomposes as a sum over training examples: $\boxed{J(X, \theta) = R(\theta) + \frac{1}{T} \sum_{i=1}^{T} l(\mathbf{x}^{(i)}, y^{(i)}, \theta)}$, where R is an optional regularization term. Stochastic Gradient Descent (SGD) approximates the gradient using a *single* randomly sampled example: $\boxed{\theta \leftarrow \theta - \eta \nabla_\theta \big(R(\theta) + l(\mathbf{x}, y, \theta)\big)}$ while Mini-batch Gradient Descent uses a small batch B of k examples: $\boxed{\theta \leftarrow \theta - \eta \nabla_\theta \left(R(\theta) + \frac{1}{k} \sum_{(\mathbf{x}, y) \in B} l(\mathbf{x}, y, \theta)\right)}$. In both cases, training proceeds over epochs (full passes over X): shuffle the dataset, iterate over single training examples or mini-batches, and update the parameters after each step. Mini-batches offer a trade-off between accurate gradients (full batch) and fast, noisy updates (SGD), and allow efficient parallel computation.

References 1/2

- Bullinaria, John A. & Joseph P. Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. Behavior Research Methods 39(3). 510–526.
- Bullinaria, John A. & Joseph P. Levy. 2012. Extracting semantic representations from word co-occurrence statistics: Stop-lists, stemming, and SVD. Behavior Research Methods 44(3). 890–907.
- Church, Kenneth W. & Patrick Hanks. 1990. Word association norms, mutual information and lexicography. Computational Linguistics 16(1). 22–29.
- Cybenko, George. 1989. Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems 2(4). 303–314.
- Deerwester, Scott C., Susan T. Dumais, Thomas K. Landauer, George W. Furnas & Richard A. Harshman. 1990. Indexing by latent semantic analysis. Journal of the American Society for Information Science 41(6). 391–407.
- Eckart, Carl & Gale Young. 1936. The approximation of one matrix by another of lower rank. Psychometrika 1(3). 211–218.
- Fano, Robert M. 1961. Transmission of information: A statistical theory of communications. Cambridge, MA: MIT Press.
- Firth, John R. 1957. A synopsis of linguistic theory, 1930–1955. In *Studies in Linguistic Analysis*, 1–32. Oxford: Basil Blackwell.

References 2/2

- Harris, Zellig S. 1954. Distributional structure. *Word* 10(2–3). 146–162.
- Hornik, Kurt, Maxwell Stinchcombe & Halbert White. 1989. Multilayer feedforward networks are universal approximators. Neural Networks 2(5). 359–366.
- Leshno, Moshe, Vladimir Ya. Lin, Allan Pinkus & Shimon Schocken. 1993. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. Neural Networks 6(6). 861–867.
- Luhn, Hans Peter. 1958. The automatic creation of literature abstracts. IBM Journal of Research and Development 2(2). 159–165.
- Minsky, Marvin & Seymour Papert. 1969. Perceptrons: An introduction to computational geometry. Cambridge, MA: MIT Press.
- Schütze, Hinrich. 1992. Dimensions of meaning. In *Proceedings of Supercomputing '92*.
- Spärck Jones, Karen. 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation 28(1). 11–21.
- Zipf, George Kingsley. 1935. The psycho-biology of language. Boston: Houghton Mifflin.