# Low-Dimensional Representation of Biological Sequence Data

Carter Tillquist

Department of Computer Science, University of Colorado Boulder
IQ Biology Program, BioFrontiers Institute

10-th ACM BCB Conference
Niagara Fall, NY

# GitHub

https://github.com/riti4538/ACM-BCB-2019-Tutorial

# Table of Contents

# Part 1: Embeddings

# Embeddings

- Sequence analysis often involves machine learning
- Many algorithms assume features lie in $\mathbb{R}^n$

# Embeddings

- Sequence analysis often involves machine learning
- Many algorithms assume features lie in $\mathbb{R}^n$
- Map symbolic data to this space
  - A naive mapping can imply unintended relationships

$$f(A) = 1 \qquad f(C) = 2 \qquad f(T) = 3 \qquad f(G) = 4$$

# Embeddings

- A map **preserving relevant structure**

$$f : X \hookrightarrow Y$$

# Embeddings

- A map **preserving relevant structure**

$$f : X \hookrightarrow Y$$

- Usually, $Y$ is a vector space ($\mathbb{R}^n$)

# Embeddings

- A map **preserving relevant structure**

$$f : X \hookrightarrow Y$$

- Usually, $Y$ is a vector space ($\mathbb{R}^n$)
- Wide variety of possibilities for $X$
  - Words, sentences, tweets, documents
  - Graphs, individual vertices
  - Images, videos
  - Audio
  - Time series
  - Multimodal
  - Sequence data

# Embeddings

$$f : X \hookrightarrow Y$$

- If $u, v \in X$ are "similar", $d(f(u), f(v))$ is small

# Embeddings

$$f : X \hookrightarrow Y$$

- If $u, v \in X$ are "similar", $d(f(u), f(v))$ is small
- Desirable properties
  - Define $f$ efficiently
  - $f(u)$ for $u \in X$ easy to compute
  - New observations are readily embeddable
  - $Y$ low-dimensional

- Graph embeddings
  - Factorization approaches: Locally Linear Embedding (LLE), Laplacian Eigenmaps, Graph Factorization, Multidimensional Scaling (MDS)
  - Deep learning: Structural Deep Network Embedding (SDNE), Graph Convolutional Networks (GCN)
  - Metric dimension and multilateration

# Embedding Techniques

## Common Techniques and Tools

- Graph embeddings
  - Factorization approaches: Locally Linear Embedding (LLE), Laplacian Eigenmaps, Graph Factorization, Multidimensional Scaling (MDS)
  - Deep learning: Structural Deep Network Embedding (SDNE), Graph Convolutional Networks (GCN)
  - Metric dimension and multilateration
- 2Vec family
  - Text: Word2Vec, Sentence2Vec, Doc2Vec, Tweet2Vec, Emoji2Vec

# Embedding Techniques

## Common Techniques and Tools

- Graph embeddings
  - Factorization approaches: Locally Linear Embedding (LLE), Laplacian Eigenmaps, Graph Factorization, Multidimensional Scaling (MDS)
  - Deep learning: Structural Deep Network Embedding (SDNE), Graph Convolutional Networks (GCN)
  - Metric dimension and multilateration
- 2Vec family
  - Text: Word2Vec, Sentence2Vec, Doc2Vec, Tweet2Vec, Emoji2Vec
  - Graphs: DeepWalk, Node2Vec, Struc2Vec, Subgraph2Vec

# Embedding Techniques

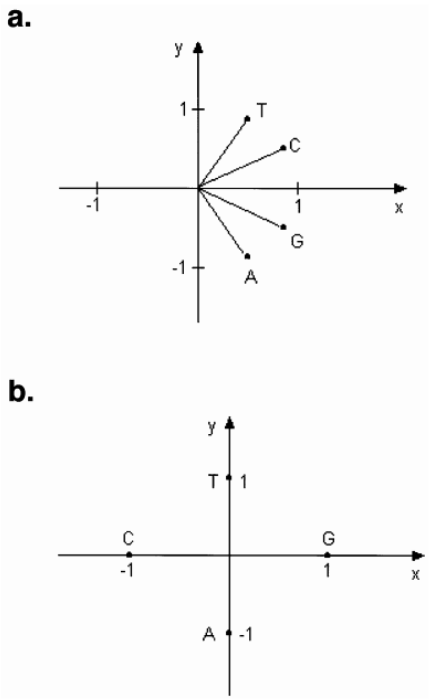
## Common Techniques and Tools

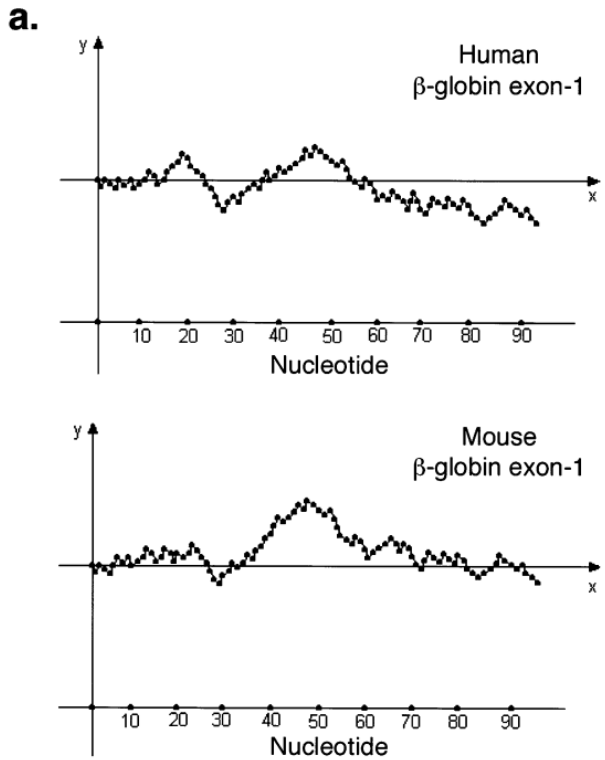
- Graph embeddings
  - Factorization approaches: Locally Linear Embedding (LLE), Laplacian Eigenmaps, Graph Factorization, Multidimensional Scaling (MDS)
  - Deep learning: Structural Deep Network Embedding (SDNE), Graph Convolutional Networks (GCN)
  - Metric dimension and multilateration
- 2Vec family
  - Text: Word2Vec, Sentence2Vec, Doc2Vec, Tweet2Vec, Emoji2Vec
  - Graphs: DeepWalk, Node2Vec, Struc2Vec, Subgraph2Vec
  - Images: Image2Vec
  - Molecules: Mol2Vec
  - Biological sequences: BioVec

# Embedding Techniques

## Biological Sequences

- $d$-dimensional representations
  - Assign each symbol (nucleotide) a vector
  - Iteratively build a curve in $\mathbb{R}^2$ [4, 14] or $\mathbb{R}^3$ [5, 11]



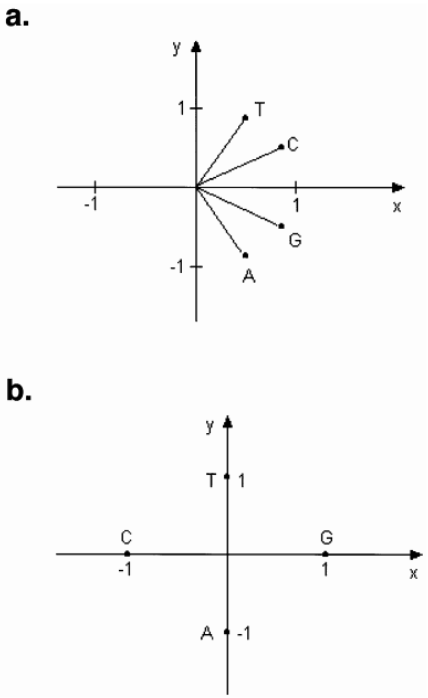



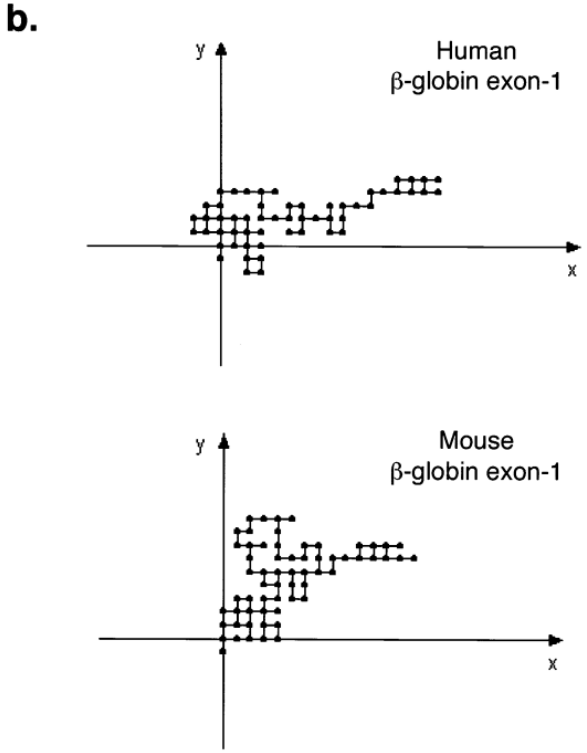

Figures from [14]

# Embedding Techniques

## Biological Sequences

- $d$-dimensional representations
  - Assign each symbol (nucleotide) a vector
  - Iteratively build a curve in $\mathbb{R}^2$ [4, 14] or $\mathbb{R}^3$ [5, 11]




Figures from [14]

# Embedding Techniques

## Biological Sequences

- $d$-dimensional representations
  - Assign each symbol (nucleotide) a vector
  - Iteratively build a curve in $\mathbb{R}^2$ [4, 14] or $\mathbb{R}^3$ [5, 11]
  - Chaos game representation [8]

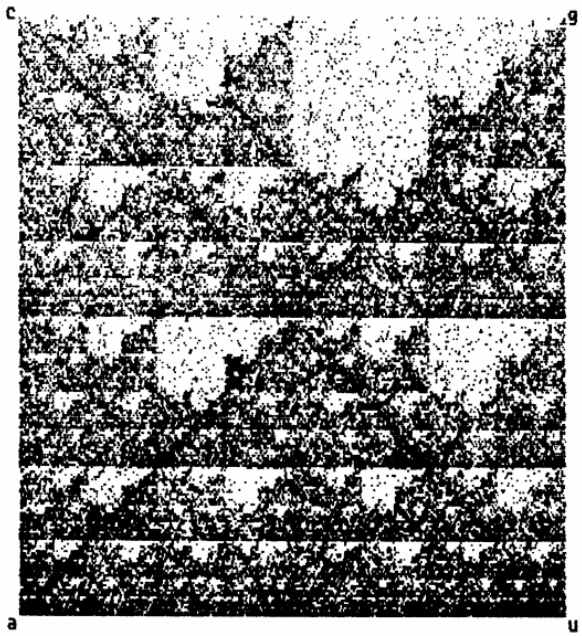



**Figure 3.** CGR of Human Beta Globin Region on Chromosome 11 (HUMHBB) (73,357 bases).

Figure from [8]

# Embedding Techniques

## Biological Sequences

- $d$-dimensional representations
  - Assign each symbol (nucleotide) a vector
  - Iteratively build a curve in $\mathbb{R}^2$ [4, 14] or $\mathbb{R}^3$ [5, 11]
  - Chaos game representation [8]
- $k$-mer count vectors
- Binary vectors
- Word2Vec/BioVec
- Multilateration

# Embedding Techniques

## $k$-mer Count Vectors

### Definition

A **k-mer** is a sequence of $k$ characters from a given alphabet.

### Definition

Using a sliding window of length $k$ over a string $S$, **k-mer count vectors** record the number of occurrences of all $|\mathcal{A}|^k$ $k$-mers from an alphabet $\mathcal{A}$.

# Embedding Techniques

## $k$-mer Count Vectors

**Definition**

A **k-mer** is a sequence of $k$ characters from a given alphabet.

**Definition**

Using a sliding window of length $k$ over a string $S$, **k-mer count vectors** record the number of occurrences of all $|\mathcal{A}|^k$ $k$-mers from an alphabet $\mathcal{A}$.

- $k = 2$, $S = AAATGGAC$

$\#(AA) = 2$ $\#(GC) = 0$

# Embedding Techniques

## $k$-mer Count Vectors

**Definition**

A **k-mer** is a sequence of $k$ characters from a given alphabet.

**Definition**

Using a sliding window of length $k$ over a string $S$, **k-mer count vectors** record the number of occurrences of all $|\mathcal{A}|^k$ $k$-mers from an alphabet $\mathcal{A}$.

- $k = 2$, $S = AAATGGAC$

$$\#(AA) = 2 \qquad \#(GC) = 0$$

$Y = (2, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0)$, ordered lexicographically

$$\dim(Y) = |\mathcal{A}|^k = 16$$

# Embedding Techniques

## Binary Vectors

### Definition

**Binary vectors** indicate the presence or absence of each character in an alphabet $\mathcal{A}$ at each position in a string.

- Concatenate one-hot encodings for every character in a sequence
- $S = AAATGGAC$

## Definition

**Binary vectors** indicate the presence or absence of each character in an alphabet $\mathcal{A}$ at each position in a string.

- Concatenate one-hot encodings for every character in a sequence
- $S = AAATGGAC$

$$A \hookrightarrow (1,0,0,0) \qquad C \hookrightarrow (0,1,0,0)$$
$$G \hookrightarrow (0,0,1,0) \qquad T \hookrightarrow (0,0,0,1)$$

$$Y = (1,0,0,0,1,0,0,0,\ldots,0,1,0,0)$$

$$\dim(Y) = |\mathcal{A}| \cdot |S| = 32$$

# Part 2: Word2Vec

# Word2Vec

## Intuition

- Embed words in $\mathbb{R}^n$
- For two words $w_1$ and $w_2$, $||f(w_1) - f(w_2)||_2$ should be small when $w_1$ and $w_2$ are "similar"

# Word2Vec

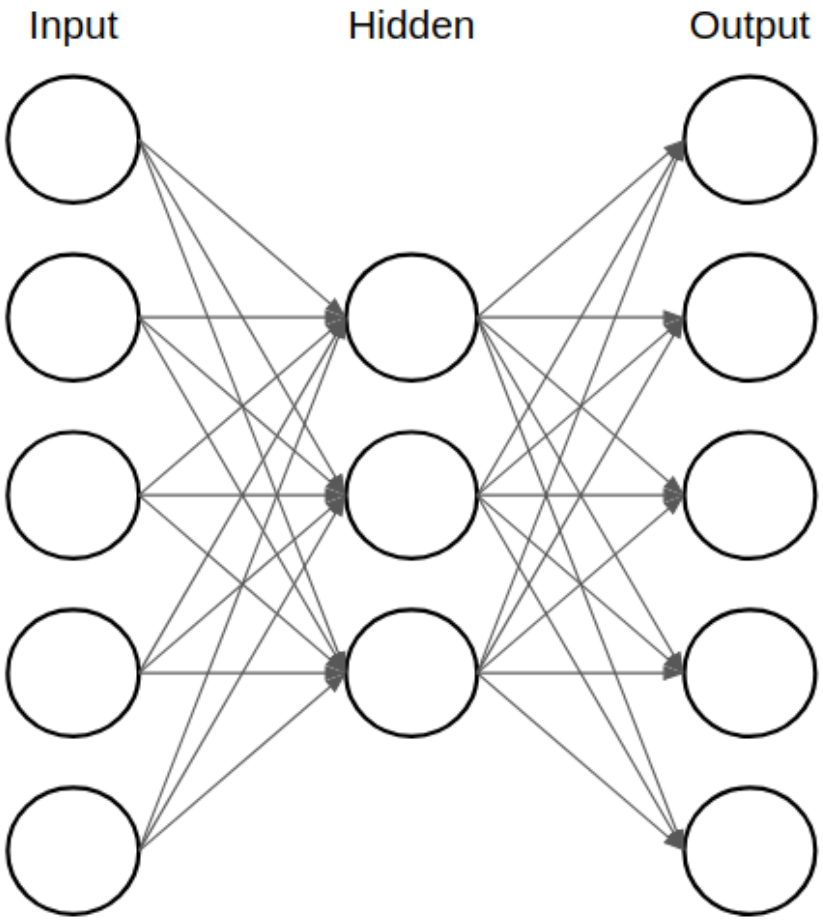
## Intuition

- Embed words in $\mathbb{R}^n$
- For two words $w_1$ and $w_2$, $||f(w_1) - f(w_2)||_2$ should be small when $w_1$ and $w_2$ are "similar"
- "Similarity" with respect to semantic meaning
- Words with similar contexts having similar meanings

# Word2Vec

## Neural Networks

- Every edge has an associated weight
  - Learned through backpropagation
- Input values move along edges
- Nodes in hidden layers include an activation function
- A softmax is often applied to the output layer



# Word2Vec

## Architecture

- Two architectures
  - Continuous skip-gram
  - Continuous bag-of-words (CBOW)

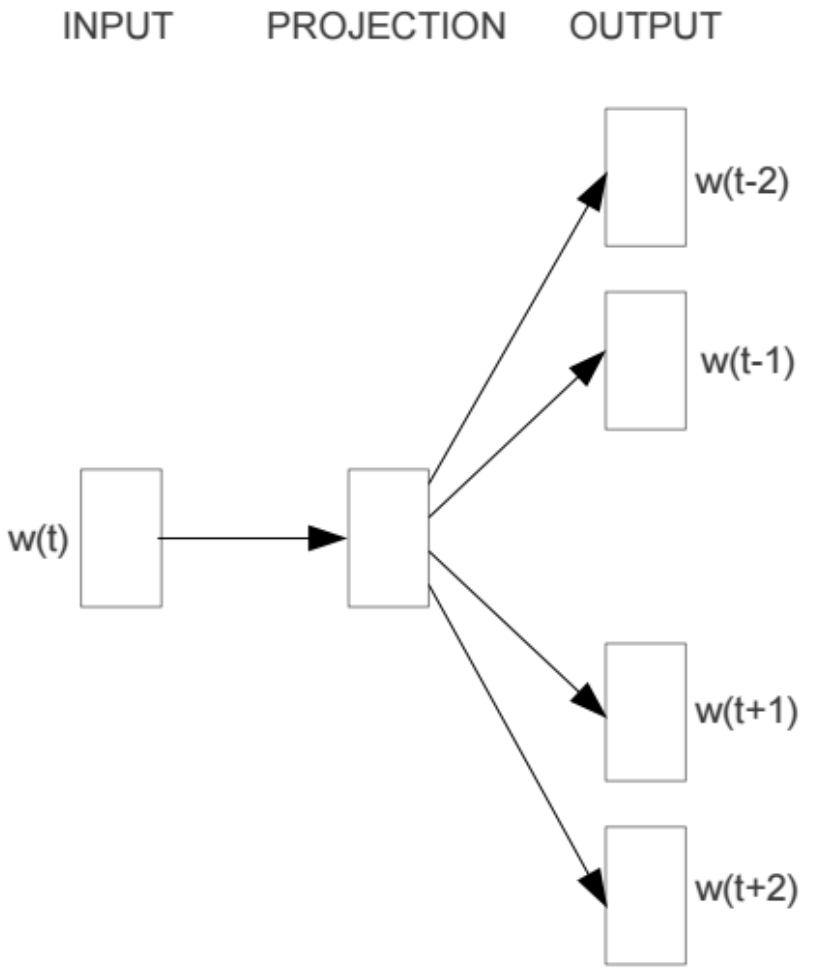



Figure from [10]

# Word2Vec

## Architecture

- Two architectures
  - Continuous skip-gram
  - Continuous bag-of-words (CBOW)
- Predict the words around $w(t)$

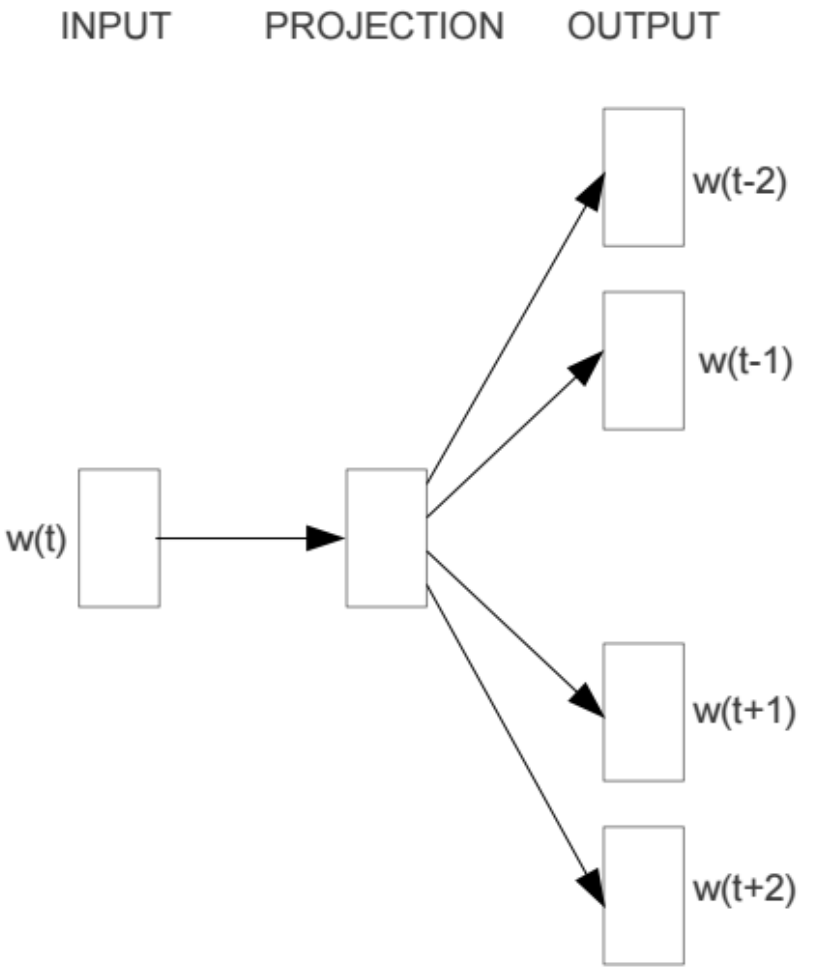



Figure from [10]

# Word2Vec

## Architecture

- Two architectures
  - Continuous skip-gram
  - Continuous bag-of-words (CBOW)
- Predict the words around $w(t)$
  - Difficult task
  - We want the weights between the input and hidden layers
  - $n$-dimensional representation of each word

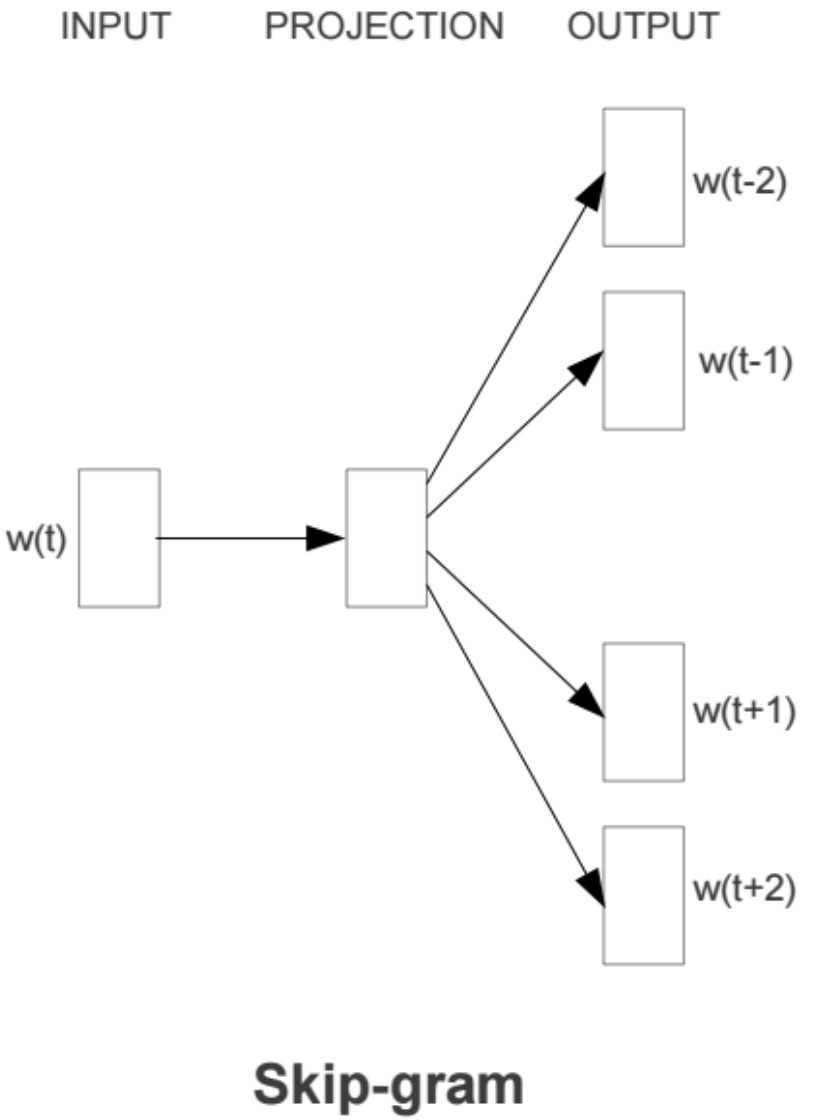



Figure from [10]

# Word2Vec

## Assessing the Embedding

- Performance on downstream task

# Word2Vec

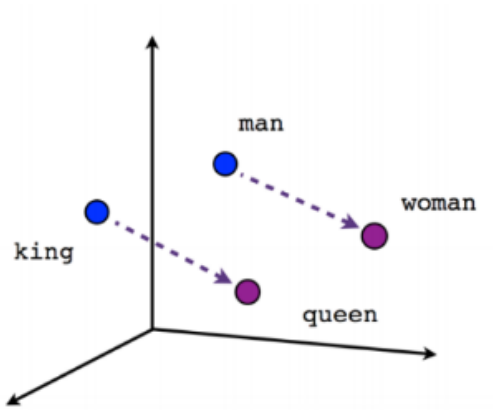
## Assessing the Embedding

- Performance on downstream task
- Analogies as vector arithmetic

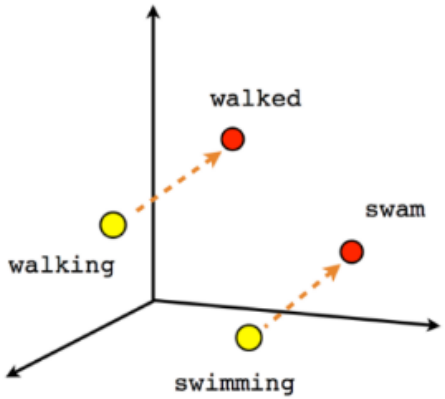
$$\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman} \approx \overrightarrow{queen}$$

- Performance on downstream task
- Analogies as vector arithmetic

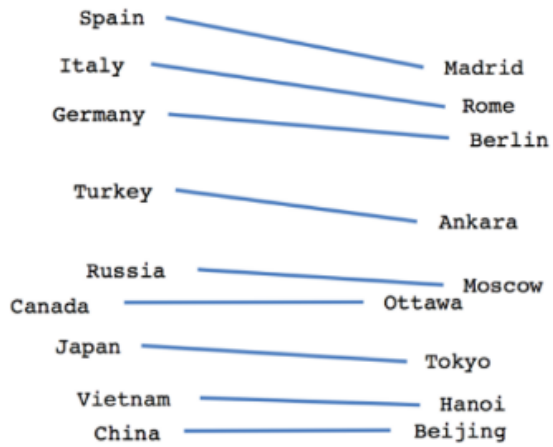
$$\overrightarrow{king} - \overrightarrow{man} + \overrightarrow{woman} \approx \overrightarrow{queen}$$




Male-Female




Verb tense




Country-Capital

https://www.tensorflow.org/tutorials/representation/word2vec

# BioVec

- BioVec for biological sequences [1]
- Embed peptide trimers in $\mathbb{R}^{100}$
- Project to two dimensions

# BioVec

## Biological Properties

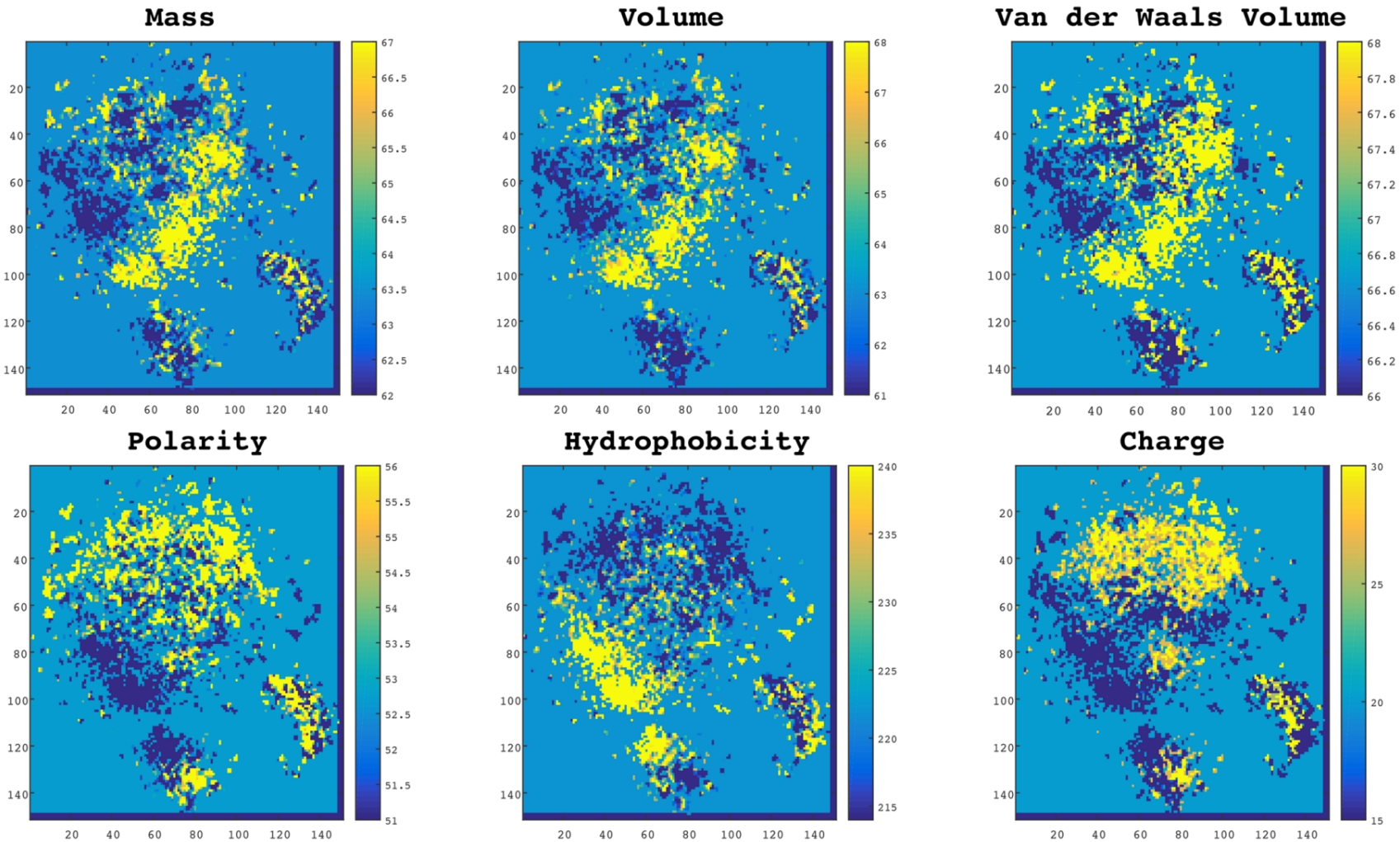



Figure from [1]

$$M(u,v) := \frac{|f(u) - f(v)|}{d(u,v)}$$

| | Lipschitz Number | | |
|---|---|---|---|
| **Property** | **protein-Space** | **The scrambled space** | **Ratio** |
| Mass | 0.3137 | 0.6605 | 0.4750 |
| Volume | 0.3742 | 0.6699 | 0.5586 |
| Van Der Waal Volume | 0.3629 | 0.6431 | 0.5643 |
| Polarity | 0.4757 | 1.2551 | 0.3790 |
| Hydrophobicity | 0.608 | 1.448 | 0.4203 |
| Charge | 0.8733 | 1.3620 | 0.6412 |
| Average | 0.50 | 1.01 | 0.51 |

Figure from [1]

- Measure the continuity of the embedding
- Embedding space is "smoother" than scrambled space

# Word2Vec

## More Details

- The dimension of the hidden layer is the embedding dimension
- Provide and predict one-hot encodings

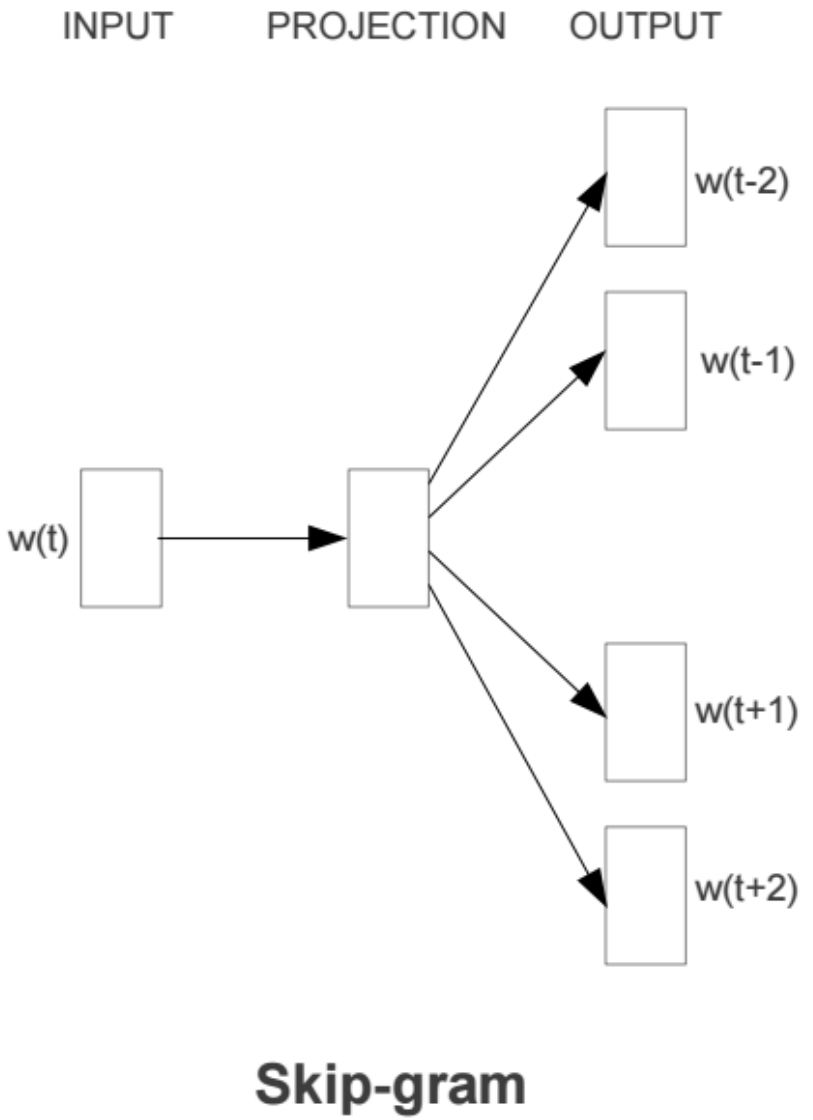



Figure from [10]

# Word2Vec

## More Details

- The dimension of the hidden layer is the embedding dimension
- Provide and predict one-hot encodings
- The network has many weights making training slow
  - Negative sampling focuses on updating the weights of a few negative examples
  - A hierarchical softmax uses a Huffman tree to reduce computation

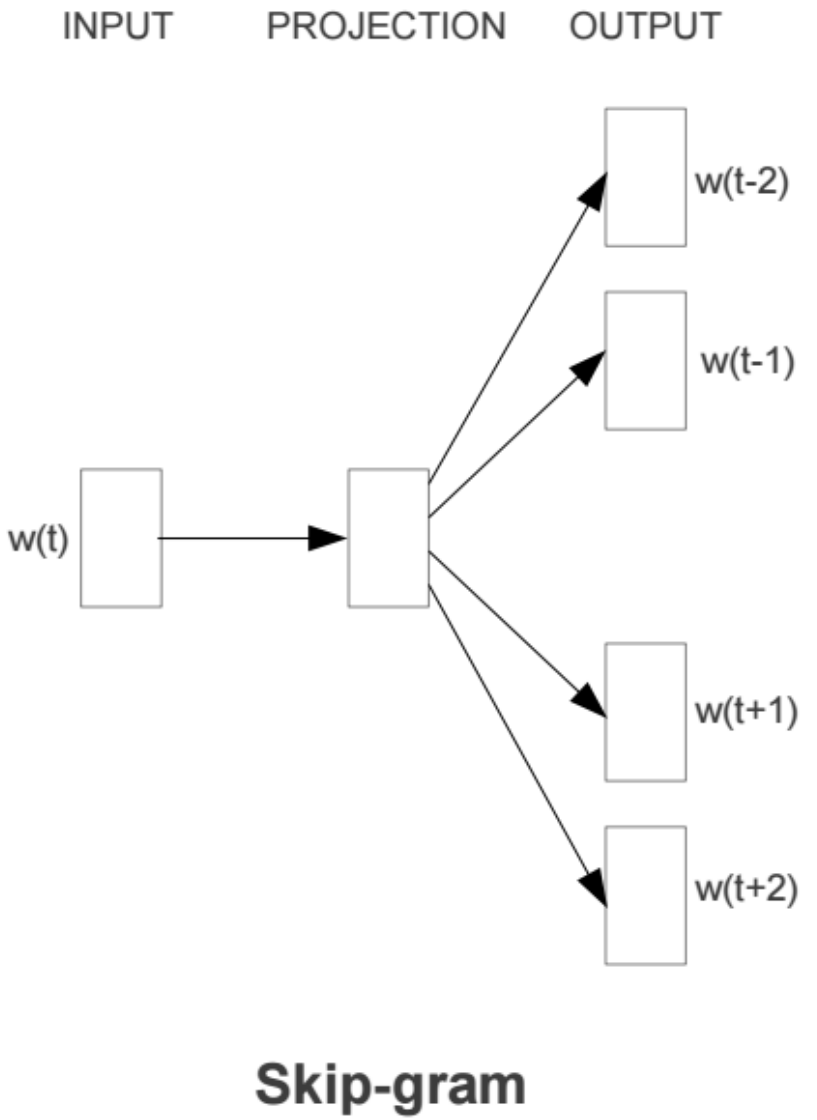



Figure from [10]

# Word2Vec

## More Details

- The dimension of the hidden layer is the embedding dimension
- Provide and predict one-hot encodings
- The network has many weights making training slow
  - Negative sampling focuses on updating the weights of a few negative examples
  - A hierarchical softmax uses a Huffman tree to reduce computation
- Words unobserved in training data can not be embedded

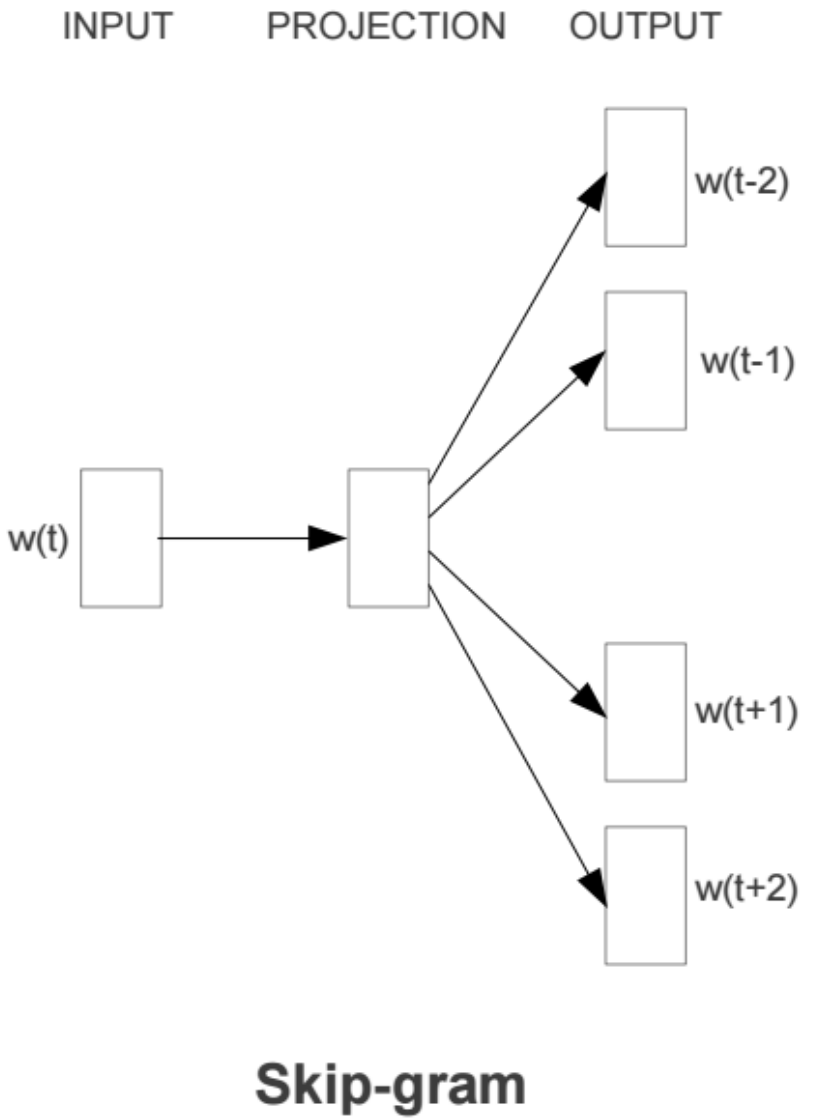



Figure from [10]

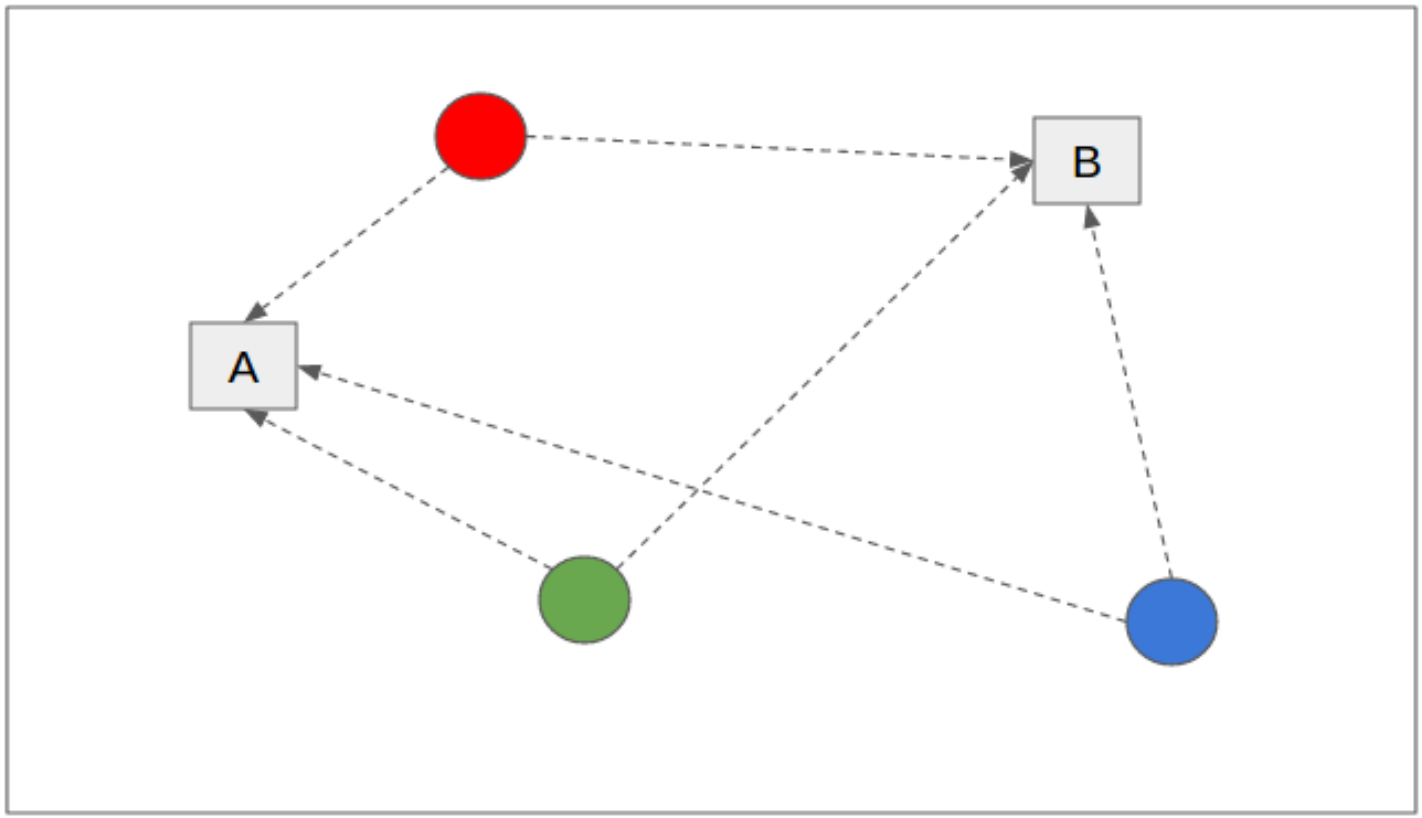
# Word2Vec Examples

https://github.com/riti4538/ACM-BCB-2019-Tutorial

# Part 3: Multilateration

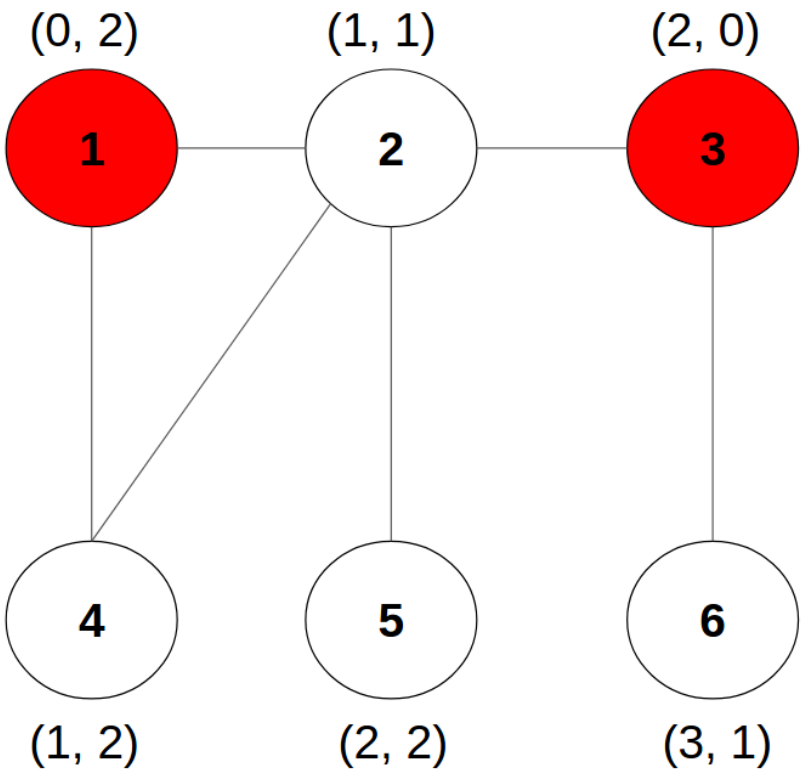
# Metric Dimension

## Trilateration



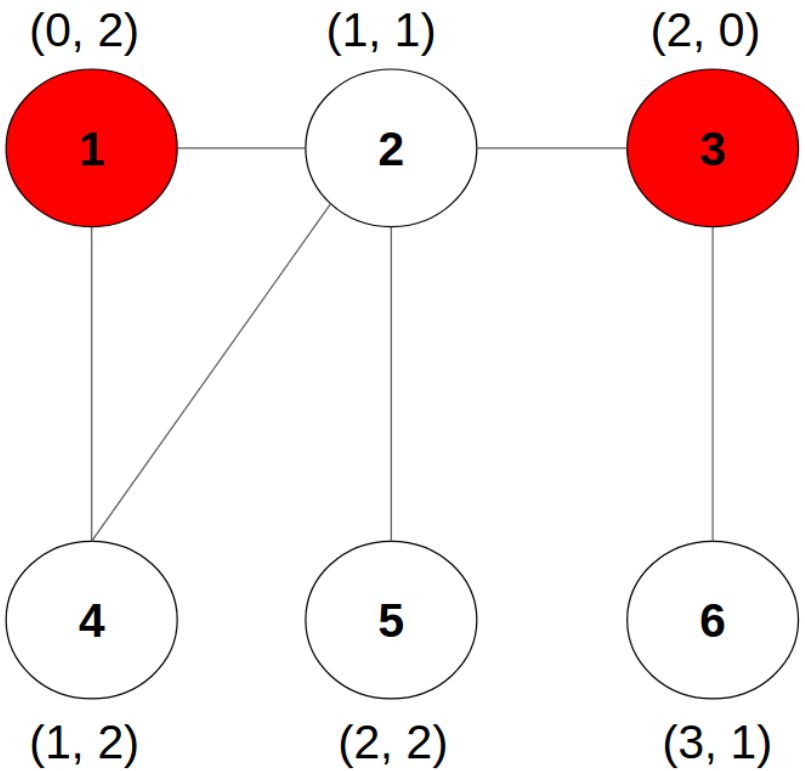
## Definition (Resolving Set)

Given a graph $G = (V, E)$, a subset of nodes $R$ is resolving if for all $u, v \in V$ there is an $r \in R$ such that $d(u, r) \neq d(v, r)$.



## Definition (Metric Dimension)

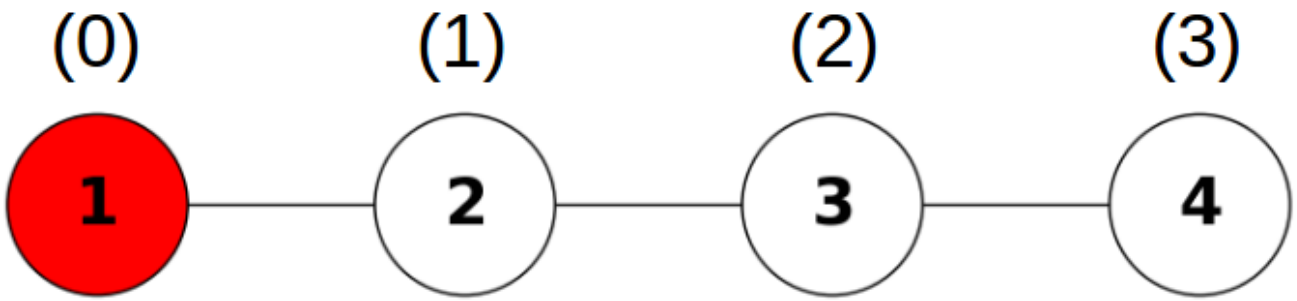
The metric dimension of a graph $G = (V, E)$, denoted $\beta(G)$, is the size of smallest resolving sets of $G$ [6, 12].



# Metric Dimension

## Path Graphs

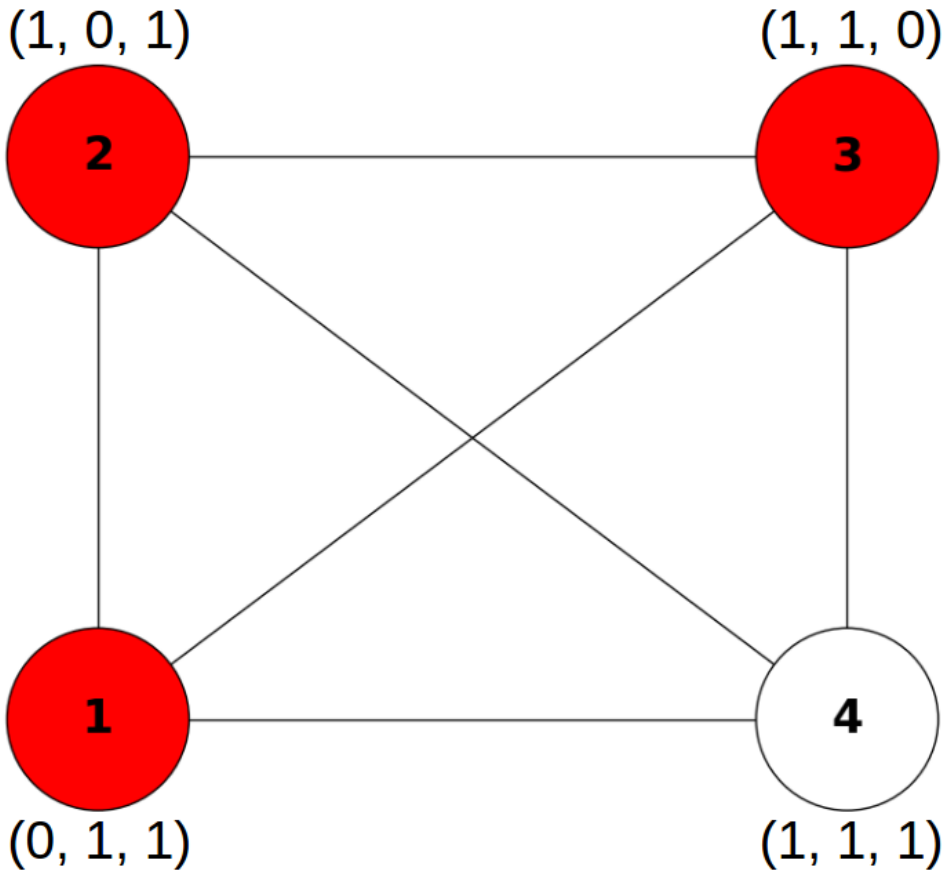
- $\beta(G) = 1$ if and only if $G$ is a path



# Metric Dimension

## Complete Graphs

- $\beta(G) = (n-1)$ if and only if $G$ is $K_n$



# Metric Dimension

## Embedding

- Given $G = (V, E)$, pick a resolving set $R$
- Represent $v \in V$ as the vector of distances to vertices in $R$

$$\Phi_R(v) = (d(v, r_1), d(v, r_2), \ldots, d(v, r_{|R|}))$$

# Metric Dimension

## Embedding

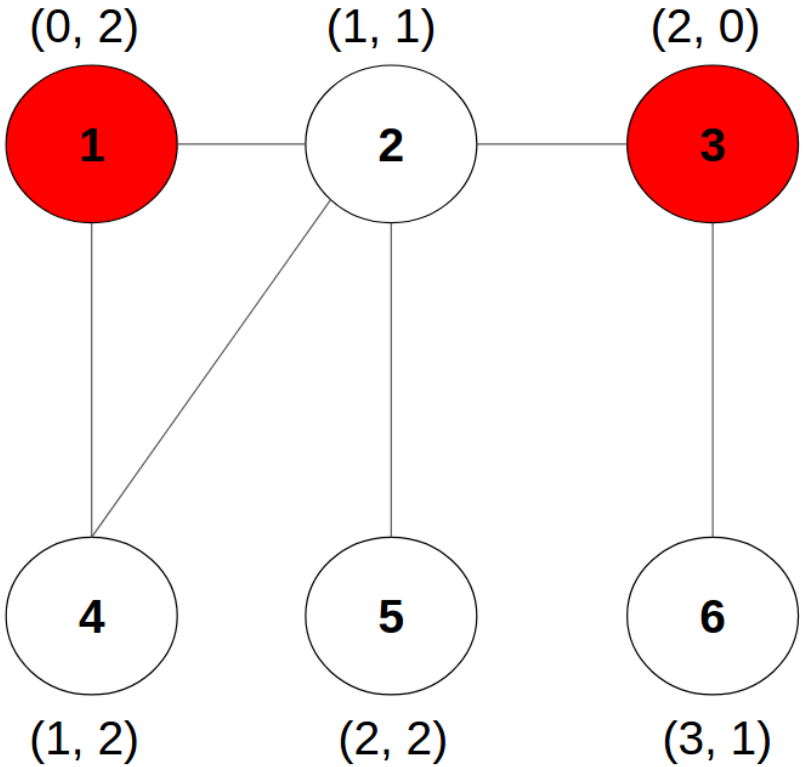
- Given $G = (V, E)$, pick a resolving set $R$
- Represent $v \in V$ as the vector of distances to vertices in $R$

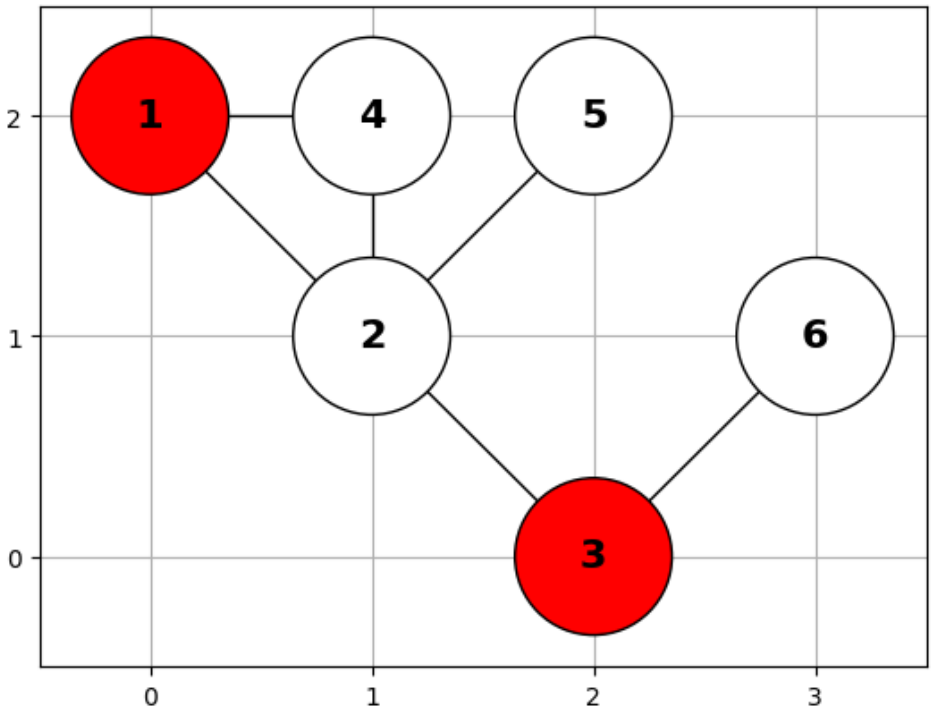
$$\Phi_R(v) = (d(v, r_1), d(v, r_2), \ldots, d(v, r_{|R|}))$$

- All vertices have a unique representation
- Intuitively, nearby vertices in $G$ will be close in the embedding

# Metric Dimension

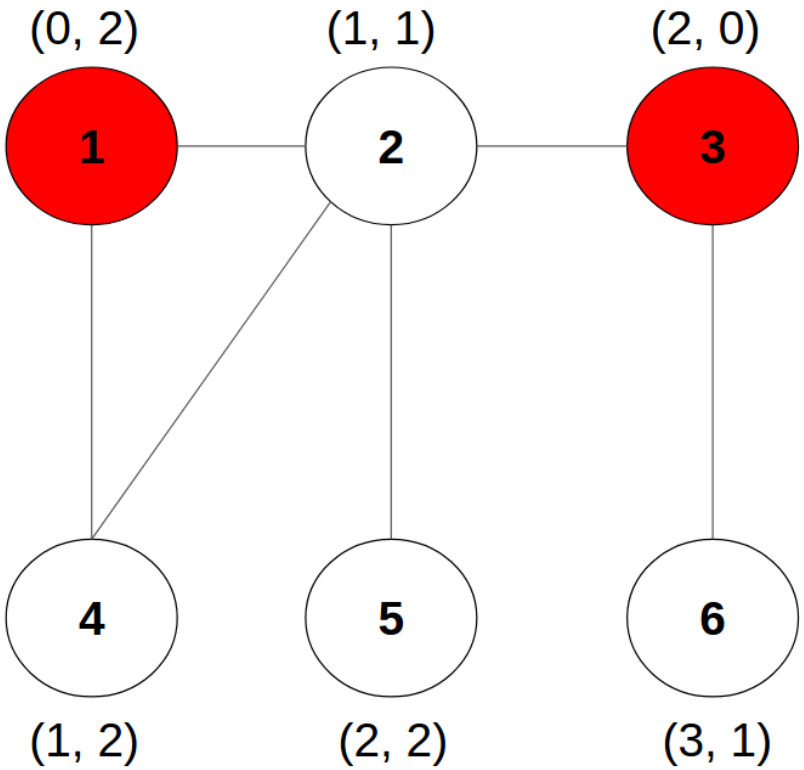
## Embedding



# Metric Dimension

## Multilateration




$$D = \begin{bmatrix} 0 & 1 & 2 & 1 & 2 & 3 \\ 1 & 0 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 2 & 2 & 1 \\ 1 & 1 & 2 & 0 & 2 & 3 \\ 2 & 1 & 2 & 2 & 0 & 3 \\ 3 & 2 & 1 & 3 & 3 & 0 \end{bmatrix}$$

$$D_R = \begin{bmatrix} 0 & 2 \\ 1 & 1 \\ 2 & 0 \\ 1 & 2 \\ 2 & 2 \\ 3 & 1 \end{bmatrix}$$

# Metric Dimension

## Complexity

- Finding $\beta(G)$ is difficult, in fact NP-complete, for general $G = (V, E)$ [3, 9]
  - A $O(|V|^3)$ approximation algorithm with approximation ratio $1 + (1 + o(1)) \ln |V|$ exists [7]
- Properties of $\beta(G)$ for specific families of graphs are known

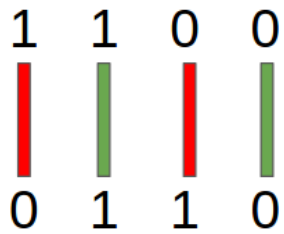
# Metric Dimension

## Complexity

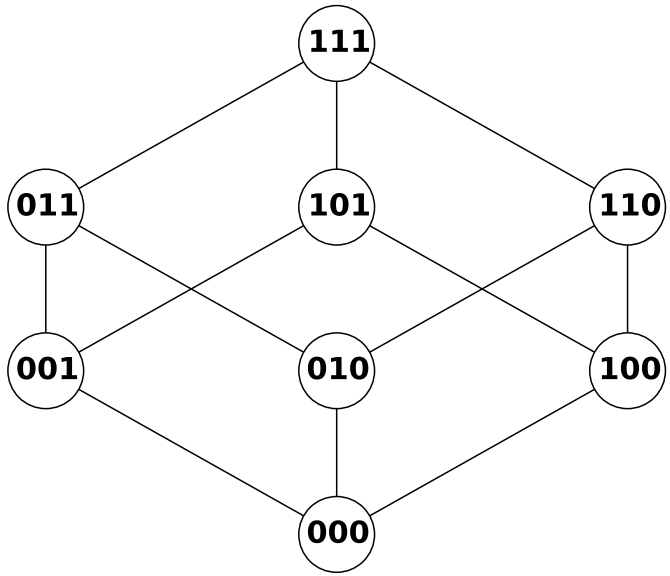
- Finding $\beta(G)$ is difficult, in fact NP-complete, for general $G = (V, E)$ [3, 9]
  - A $O(|V|^3)$ approximation algorithm with approximation ratio $1 + (1 + o(1)) \ln |V|$ exists [7]
- Properties of $\beta(G)$ for specific families of graphs are known
- $\beta(G_{n,p}) = O(\frac{-2\ln(n)}{\ln(p^2+(1-p)^2)})$ across many regimes of $p$ [2]
- $\beta(G)$ can be determined exactly in polynomial time for trees [6, 12]

# Hamming Graphs

- The Hamming distance counts mismatches between strings




- For $H_{k,a}$
  - $V$ is the set of all $k$-mers from an alphabet of size $a$
  - $(u, v) \in E$ when strings $u$ and $v$ differ in exactly one position




$H_{3,2}$

**Theorem (Tillquist and Lladser [13])**

$$\beta(H_{k,a}) \leq \beta(H_{k+1,a}) \leq \beta(H_{k,a}) + \lfloor \frac{a}{2} \rfloor$$

**Theorem (Tillquist and Lladser [13])**

$$\beta(H_{k,a}) \leq \beta(H_{k+1,a}) \leq \beta(H_{k,a}) + \lfloor \frac{a}{2} \rfloor$$

- $H_{k,a}$ can be embedded in $O(k)$-dimensions
- The proof is constructive

$$D = \begin{array}{c} \\ V_0 \\ V_1 \\ \vdots \\ V_{a-1} \end{array} \begin{array}{c} \begin{array}{cccc} V_0 & V_1 & \cdots & V_{a-1} \end{array} \\ \begin{pmatrix} d & d+\mathbf{1} & \cdots & d+\mathbf{1} \\ d+\mathbf{1} & d & \cdots & d+\mathbf{1} \\ \vdots & \vdots & \ddots & \vdots \\ d+\mathbf{1} & d+\mathbf{1} & \cdots & d \end{pmatrix} \end{array}$$

# Hamming Graphs

## $H_{8,20}$

- Consider $H_{8,20}$
  - 25.6 billion vertices
  - More than six hundred quintillion entries in the distance matrix

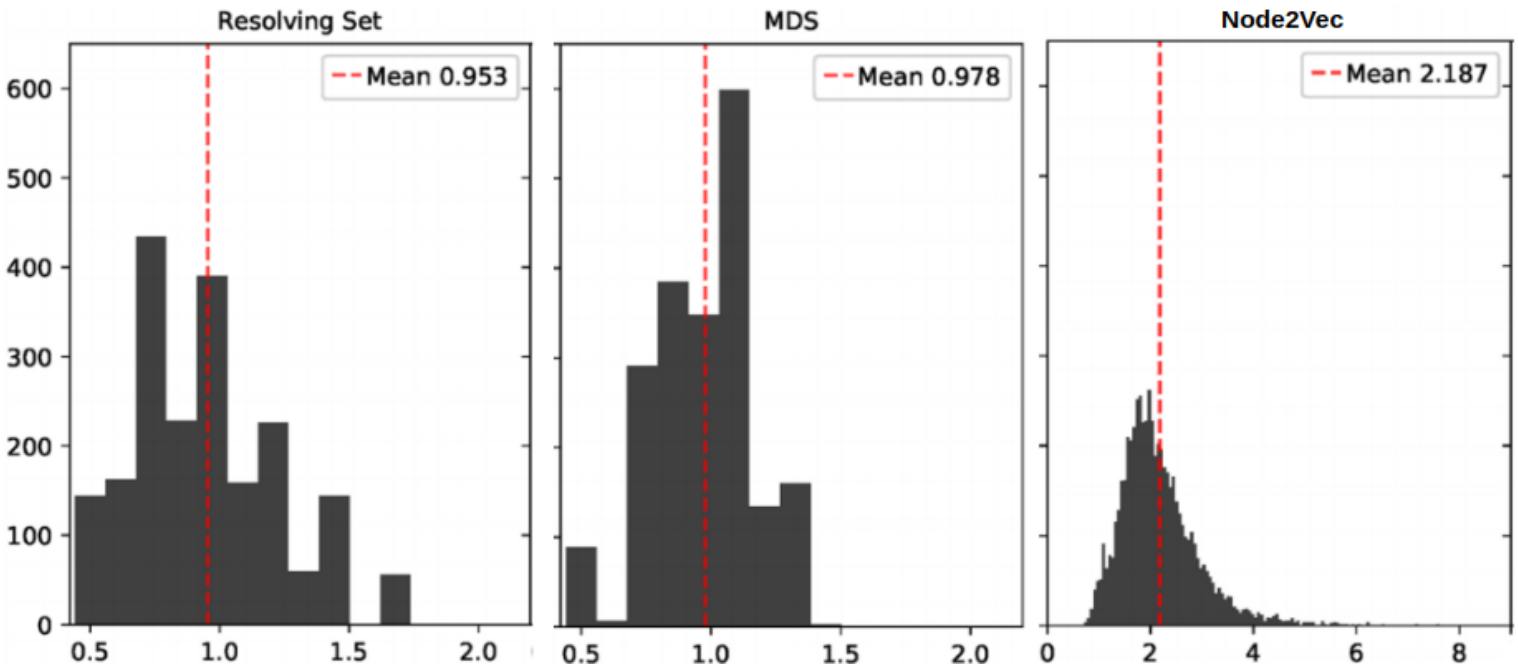
# Hamming Graphs

## $H_{8,20}$

- Consider $H_{8,20}$
  - 25.6 billion vertices
  - More than six hundred quintillion entries in the distance matrix
- $\beta(H_{3,20}) \leq 32$ via ICH approximation
- $\beta(H_{8,20}) \leq 82$ via construction
- 3-mer count vector: $20^3 = 8000$ dimensions (97.56 times bigger)
- Binary vectors: $20 \times 8 = 160$ dimensions (1.95 times bigger)

# Hamming Graphs

## Metric Distortion of $H_{3,4}$

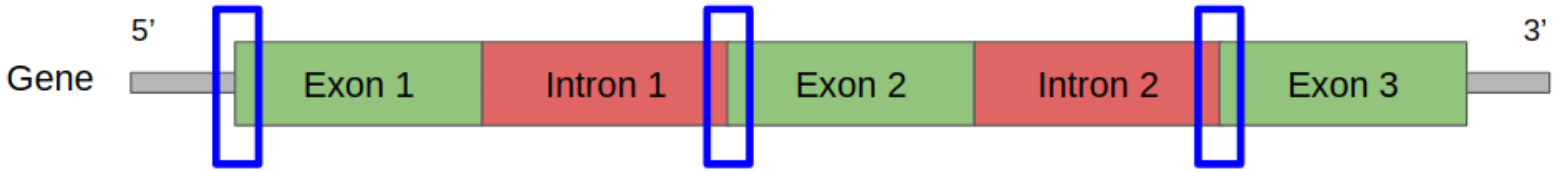



$$M(u,v) := \frac{d(u,v)}{||f(u) - f(v)||_2}$$

- $d(\cdot,\cdot)$ is the Hamming distance
- $f : \{A, C, G, T\}^3 \to \mathbb{R}^6$ is the embedding

# Proof-of-Concept

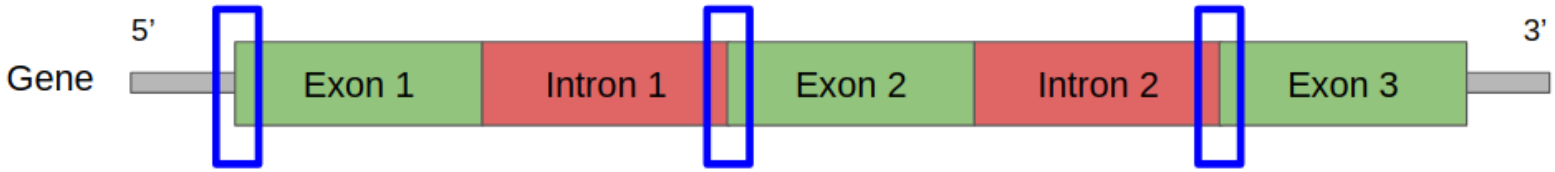
## Task




- Extract 20-mers from the *Drosophila melanogaster* genome
- Classify as being centered or not at exon start sites

# Proof-of-Concept

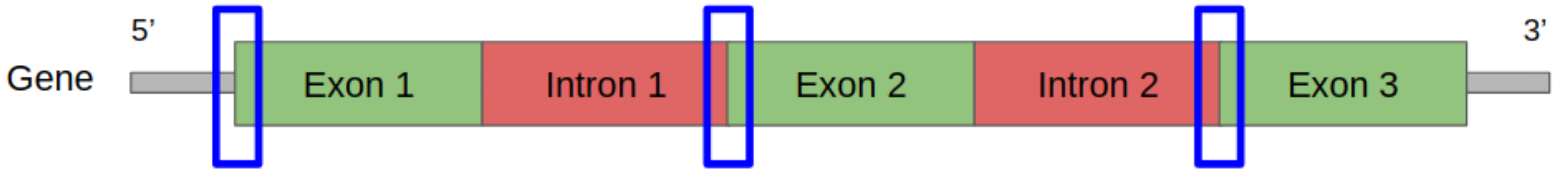
## Task




- Extract 20-mers from the *Drosophila melanogaster* genome
- Classify as being centered or not at exon start sites
- Data
  - $\sim$ 87 thousand positive examples
  - $\sim$ 87 thousand negative examples drawn from the genome

# Proof-of-Concept

## Task

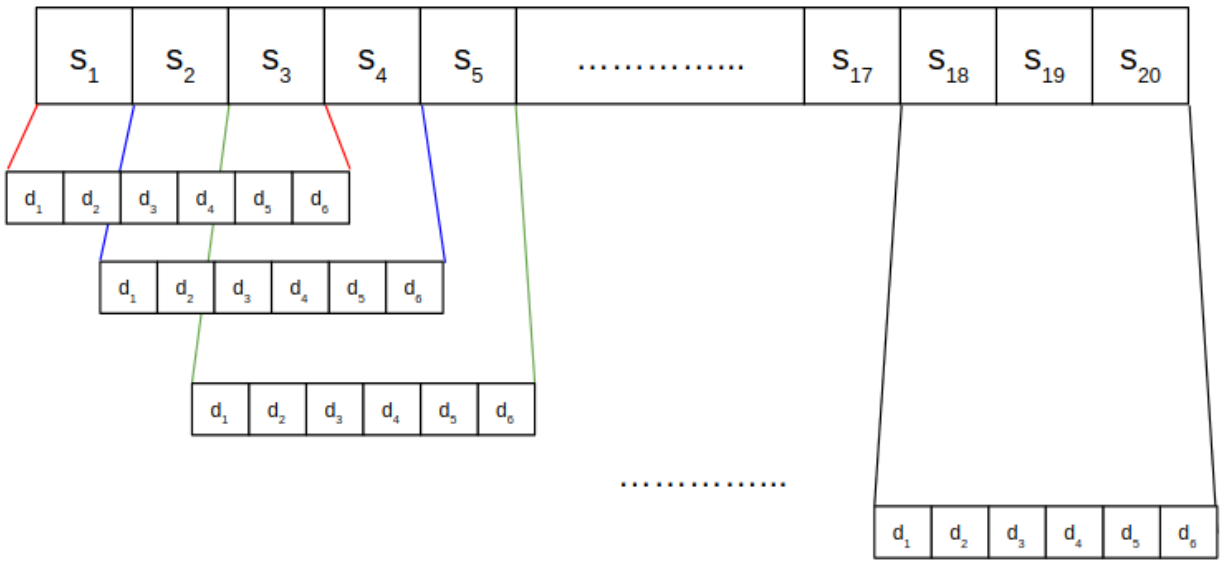



- Extract 20-mers from the *Drosophila melanogaster* genome
- Classify as being centered or not at exon start sites
- Data
  - $\sim$ 87 thousand positive examples
  - $\sim$ 87 thousand negative examples drawn from the genome
- Features
  - Metric dimension, multidimensional scaling, Node2Vec
  - 3-mer count and binary vectors

# Proof-of-Concept

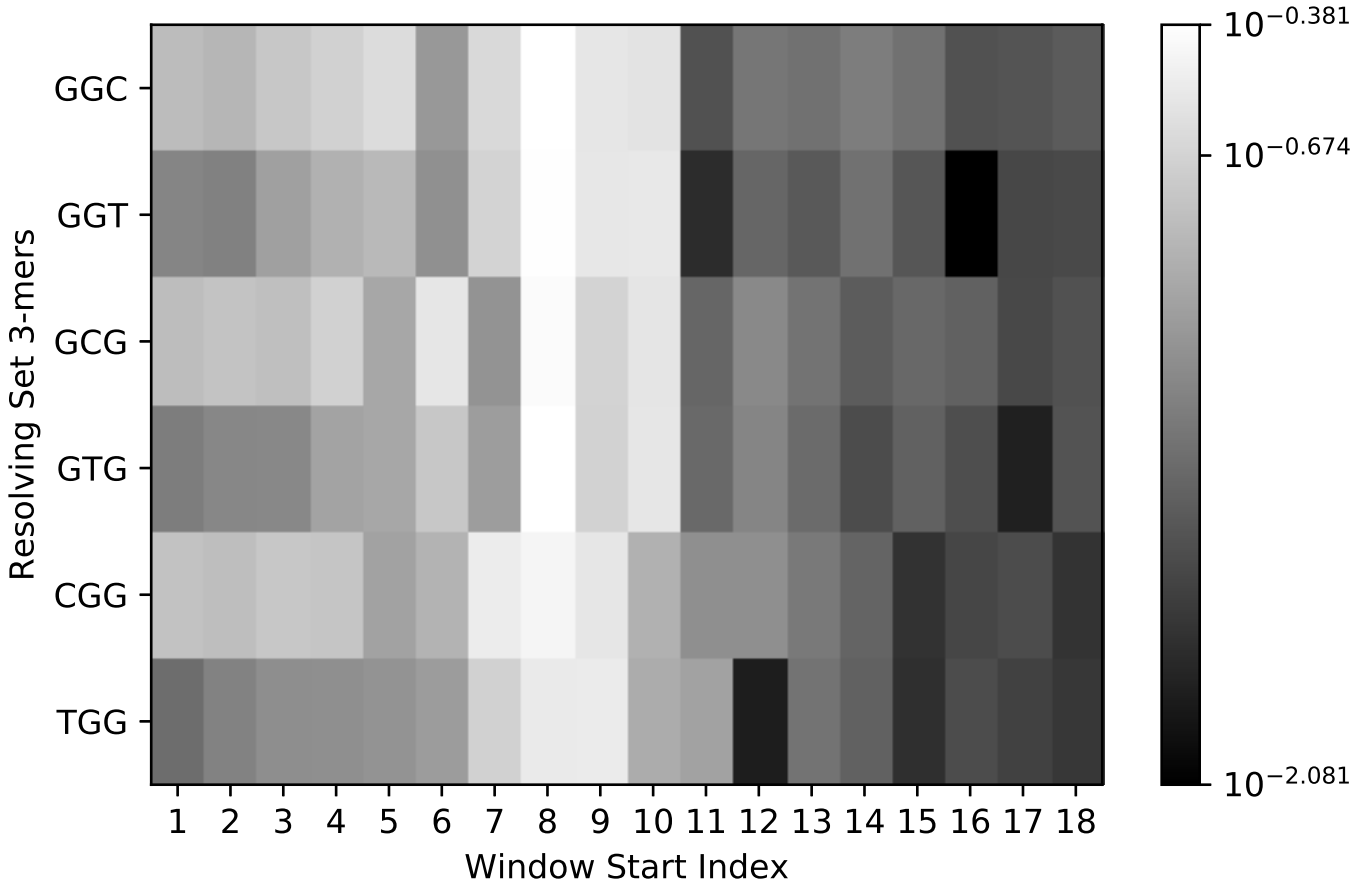
## Metric Dimension Features

- Focus on 3-mers
  - Embed the nodes of $H_{3,4}$ in $\mathbb{R}^6$
  - Use a sliding window of length 3
  - 108 total features



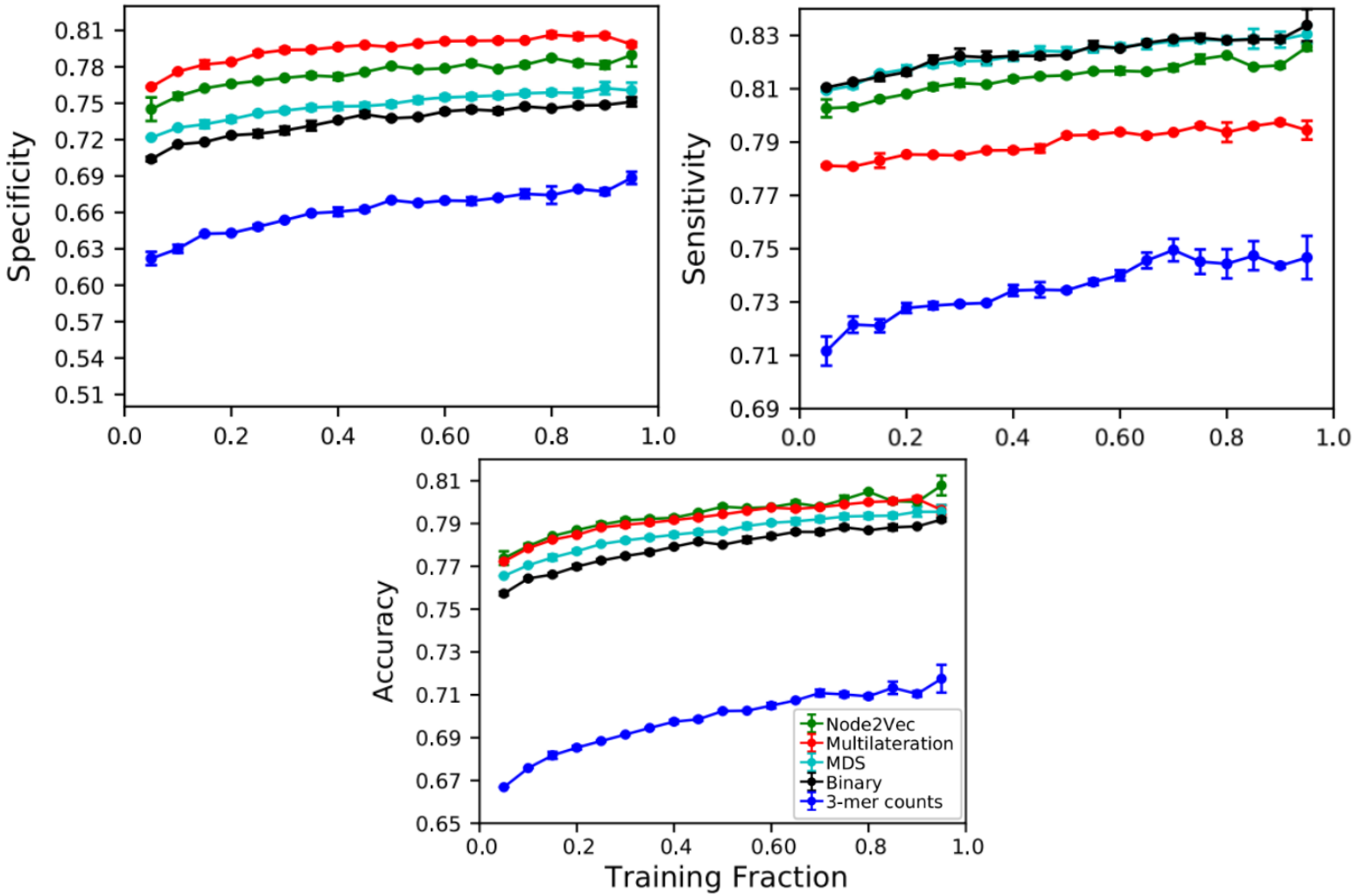
# Proof-of-Concept

## Positive vs Negative Examples



# Proof-of-Concept

## Comparisons



# Multilateration Example

https://github.com/riti4538/ACM-BCB-2019-Tutorial

# Conclusion

- Embedding symbolic data in $\mathbb{R}^n$ is often important for analysis
- Embeddings can be generated in many different ways
- The choice of embedding can affect the quality of an analysis

# Acknowledgements


- M. Lladser (advisor)
- NSF IIS grant 1836914 (PI: Lladser)
- BioFrontiers Computing Core (NIH 1S10OD012300)
- IQ Biology (NSF IGERT grant 1144807)


# References I


[1] Asgari, E., and Mofrad, M. R.
Continuous distributed representation of biological sequences for deep proteomics and genomics.
*PloS one 10*, 11 (2015), e0141287.

[2] Bollobas, B., Mitsche, D., and Pralat, P.
Metric dimension for random graphs.
*The Electronic Journal of Combinatorics 20*, 4 (2013).

[3] Gary, M. R., and Johnson, D. S.
*Computers and Intractability: A Guide to the Theory of NP-completeness*.
WH Freeman and Company, New York, 1979.

[4] Gates, M. A.
A simple way to look at DNA.
*Journal of Theoretical Biology 119*, 3 (1986), 319–328.

# References II


[5] Hamori, E., and Ruskin, J.
H curves, a novel method of representation of nucleotide series especially suited for long DNA sequences.
*Journal of Biological Chemistry 258*, 2 (1983), 1318–1327.

[6] Harary, F., and Melter, R.
On the metric dimension of a graph.
*Ars Combinatoria 2*, 1 (1976), 191–195.

[7] Hauptmann, M., Schmied, R., and Viehmann, C.
Approximation complexity of metric dimension problem.
*Journal of Discrete Algorithms 14* (2012), 214–222.

[8] Jeffrey, H. J.
Chaos game representation of gene structure.
*Nucleic acids research 18*, 8 (1990), 2163–2170.

# References III


[9] Khuller, S., Raghavachari, B., and Rosenfeld, A.
Landmarks in graphs.
*Discrete Applied Mathematics 70*, 3 (1996), 217–229.

[10] Mikolov, T., Chen, K., Corrado, G., and Dean, J.
Efficient estimation of word representations in vector space.
*arXiv preprint arXiv:1301.3781* (2013).

[11] Randić, M., Vracko, M., Nandy, A., and Basak, S. C.
On 3-d graphical representation of DNA primary sequences and their numerical characterization.
*Journal of chemical information and computer sciences 40*, 5 (2000), 1235–1244.

[12] Slater, P. J.
Leaves of trees.
*Congressus Numerantium 14*, 549-559 (1975), 37.

# References IV


[13] Tillquist, R. C., and Lladser, M. E.
Low-dimensional representation of genomic sequences.
*Journal of mathematical biology*, 1–29.

[14] Yau, S. S.-T., Wang, J., Niknejad, A., Lu, C., Jin, N., and Ho, Y.-K.
DNA sequence representation without degeneracy.
*Nucleic acids research 31*, 12 (2003), 3078–3080.