How embeddings represent words in vector space

Embeddings represent words in vector space by assigning each word a fixed-length numerical vector. These vectors capture semantic meaning, enabling similar words to have similar vector representations. This representation allows models to process text more effectively by leveraging the geometric relationships between vectors. Here's how embeddings work in detail:

1. High-Dimensional Representation

- Each word is represented as a dense vector in a high-dimensional space.
- For example, in a 300-dimensional embedding space, each word is a 300-element vector.

2. Semantic Proximity

- Words with similar meanings or contexts (e.g., "king" and "queen") are closer to each other in the vector space.
- This is achieved by training embeddings to minimize the distance (e.g., cosine similarity) between vectors of words that occur in similar contexts.

3. Dimensionality Reduction

- Word embeddings map words from a sparse, high-dimensional space (e.g., one-hot encoding) into a dense, lower-dimensional space.
- This compact representation makes computations efficient and relationships more explicit.

4. Contextual Information

- Traditional embeddings like Word2Vec and GloVe produce a single vector for each word, capturing general semantic meaning.
- Contextual embeddings (e.g., from models like BERT) generate dynamic vectors that depend on the word's usage in a specific sentence, handling polysemy (e.g., "bank" as a financial institution vs. riverbank).

5. Training Process

- Word2Vec: Predicts a word based on its context (CBOW) or predicts the context given a word (Skip-gram).
- GloVe: Captures word co-occurrence statistics in a global corpus.

6. Geometric Relationships

- Word embeddings can represent linguistic patterns through vector arithmetic:
    - King – Man + Woman ≈ Queen
- These relationships arise because the training process encodes syntactic and semantic regularities.

Visualization

In a 2D or 3D projection of the embedding space (e.g., PCA), clusters of similar words and patterns reflecting analogies (like verb tenses, gender differences, etc.) become visible.

Reducing Dimensions and Visualizing Vectors

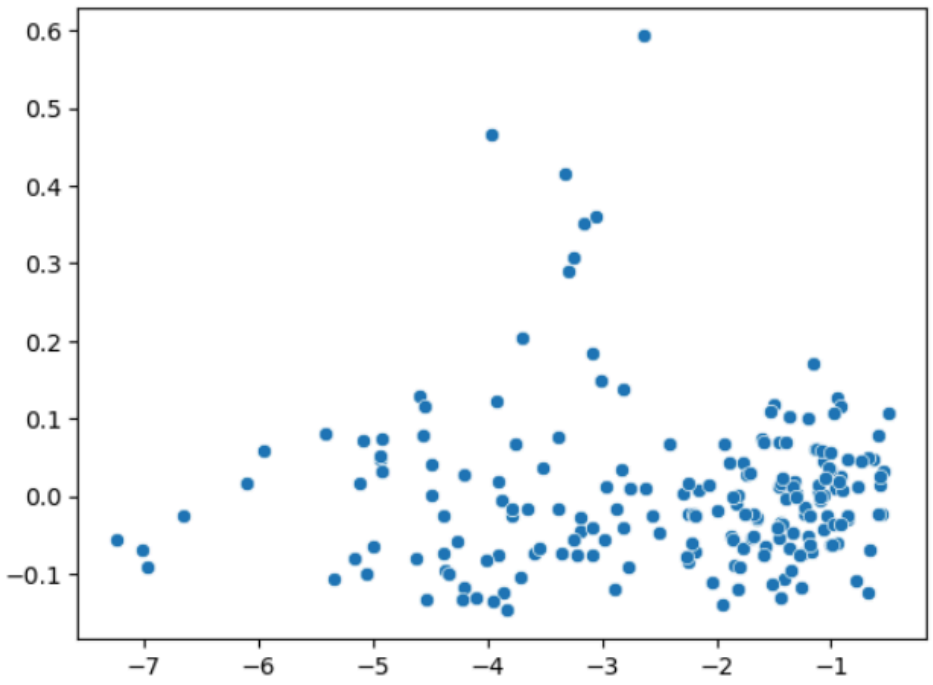
```
pca = PCA(n_components = 2)
```

```
vectors = [w2v.wv[word] for word in w2v.wv.index_to_key]
```

```
flattened_words = pca.fit_transform(vectors)
```

```
sns.scatterplot(x = flattened_words[:200, 0], y = flattened_words[:200, 1])
```

```
<Axes: >
```



Interactive Visualization using Plotly

```
import plotly.io as pio
import plotly.graph_objects as go
```

```
scatter = go.Scatter(x = df['X_Dim'][:150], y = df['Y_Dim'][:150], text = df['Words'], mode = 'markers+text',
                     textposition = 'top left', marker = dict( size = 5 , color = 'rgba(157,0,0, .8)'))
```

```
layout = go.Layout(title = 'Representing Vectors', xaxis = dict(title = 'X Dimension'),
                   yaxis = dict(title = 'Y Dimention'), hovermode = 'closest')
```

```
fig = go.Figure(data = [scatter], layout = layout)
```

```
pio.show(fig)
```

Pre-trained Word Embedding Models

GloVe (Global Vectors for Word Representation)

Models:

1. **glove-wiki-gigaword-50 :**
    a. Dimensions: 50
    b. Corpus: Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab).
    c. Use Case: Lightweight tasks where small vector size is sufficient, e.g., quick text classification.
    d. Advantages: Low memory footprint and fast computation.

```
wiki_50_glove = api.load('glove-wiki-gigaword-50')
```

```
wiki_50_glove.most_similar('food', topn = 10)
```

```
[('products', 0.7984759211540222),
 ('coffee', 0.7853199243545532),
 ('supplies', 0.7699311375617981),
 ('meat', 0.769260823726654),
 ('foods', 0.7517945766448975),
 ('supply', 0.7429676651954651),
 ('goods', 0.7426424026489258),
 ('items', 0.7383918166160583),
 ('vegetables', 0.7383119463920593),
 ('medicines', 0.7366275787353516)]
```

2. **glove-wiki-gigaword-200 :**
    a. Dimensions: 200
    b. Corpus: Same as above.
    c. Use Case: More complex tasks like sentiment analysis or basic NLP tasks where moderate semantic detail is required.

```
wiki_200_glove = api.load('glove-wiki-gigaword-200')
```

```
wiki_200_glove.most_similar('food', topn = 10)
```

```
[('foods', 0.7071868181228638),
 ('supplies', 0.6728929877281189),
 ('products', 0.6317433714866638),
 ('eating', 0.6229158639907837),
 ('eat', 0.6227813363075256),
 ('meat', 0.6220186352729797),
 ('vegetables', 0.6154162883758545),
 ('foodstuffs', 0.6085902452468872),
 ('meals', 0.6050760746002197),
 ('nutrition', 0.6017112731933594)]
```

3. **glove-wiki-gigaword-300 :**
    a. Dimensions: 300
    b. Corpus: Same as above.
    c. Use Case: Tasks requiring richer semantic relationships and analogies, e.g., document similarity and machine learning models for NLP.
    d. Advantages: Captures more nuanced relationships between words due to higher dimensionality.

```
wiki_300_glove = api.load('glove-wiki-gigaword-300')
```

```
wiki_300_glove.most_similar('food', topn = 10)
```

```
[('foods', 0.6694071292877197),
 ('supplies', 0.6068812608718872),
 ('eat', 0.5903003215789795),
 ('meat', 0.5677605271339417),
 ('eating', 0.5674231052398682),
 ('meals', 0.5671917200088501),
 ('products', 0.5494305491447449),
 ('meal', 0.5464715957641602),
 ('vegetables', 0.5447412133216858),
 ('nutrition', 0.5385703444480896)]
```

Key Characteristics of GloVe:

- Training Objective: Optimizes word co-occurrence probabilities in the corpus.
- Strengths:
    - Captures global statistical information about the corpus.
    - Pre-trained embeddings available in multiple dimensions.
- Weaknesses:
    - Static embeddings: A word has the same vector regardless of context (e.g., "bank" in finance vs. riverbank).

Word2Vec

Model:

word2vec-google-news-300:

- Dimensions: 300
- Corpus: Google News dataset (100 billion words).
- Use Case: Tasks requiring detailed semantic and syntactic information, such as analogy solving and advanced NLP applications.
- Advantages: Comprehensive coverage of word relationships and high-quality embeddings.

Key Characteristics of Word2Vec:

- Training Objective:
  - Skip-gram: Predicts surrounding words given the current word.
  - CBOW (Continuous Bag of Words): Predicts the current word given its surrounding words.
- Strengths:
  - Focuses on local context (sliding window approach).
  - Produces high-quality embeddings for syntactic and semantic relationships.
- Weaknesses:
  - Similar to GloVe, embeddings are static and context-independent.

```
google_news_300_w2v.most_similar('food', topn = 10)
```

```
[('foods', 0.6804922819137573),
 ('Food', 0.6538903713226318),
 ('foodstuffs', 0.642582893371582),
 ('meals', 0.616668701171875),
 ('food_stuffs', 0.5928642153739929),
 ('nourishing_meals', 0.5847609043121338),
 ('foodstuff', 0.5835224986076355),
 ('staple_foods', 0.5535788536071777),
 ('nutritious', 0.5466451644897461),
 ('meal', 0.5433712601661682)]
```

```
google_news_300_w2v.most_similar('war', topn = 10)
```

```
[('wars', 0.748465895652771),
 ('War', 0.6410670280456543),
 ('invasion', 0.5892109870910645),
 ('Persian_Gulf_War', 0.5890660285949707),
 ('Vietnam_War', 0.5886474847793579),
 ('Iraq', 0.5885995030403137),
 ('unwinnable_quagmire', 0.5681803226470947),
 ('un_winnable', 0.5606350302696228),
 ('occupation', 0.5506216883659363),
 ('conflict', 0.5506188869476318)]
```

Model	Dimensions	Corpus	Size	Context-Dependent	Use Cases
`glove-wiki-gigaword-50`	50	Wikipedia + Gigaword (6B tokens)	Small	No	Lightweight NLP tasks.
`glove-wiki-gigaword-200`	200	Wikipedia + Gigaword	Medium	No	Sentiment analysis, entity recognition.
`glove-wiki-gigaword-300`	300	Wikipedia + Gigaword	Large	No	Text similarity, document classification.
`word2vec-google-news-300`	300	Google News (100B words)	Large	No	Syntactic and semantic analysis.