Mathematical structures for word embeddings

Siddharth Bhat

IIIT Hyderabad

October 23th, 2021

What is a word embedding?

What is a word embedding?

- Map words to *mathematical objects*.

What is a word embedding?

- Map words to *mathematical objects*.
- Semantic ideas on words $\simeq$ mathematical operations on these objects.

What is a word embedding?

- Map words to *mathematical objects*.
- Semantic ideas on words $\simeq$ mathematical operations on these objects.
- Most common: *vector embeddings* (`word2vec`)

What is a word embedding?

- Map words to *mathematical objects*.
- Semantic ideas on words $\simeq$ mathematical operations on these objects.
- Most common: *vector embeddings* (`word2vec`)

What's word2vec?

```
def train(corpus: list, DIMSIZE: int):
  """
  train word2vec of dimension DIMSIZE on the given corpus (list of words).
  Eg:train(["the", "man", "was" "tall", "the", "quick", "brown", "fox"], 20)
  """
  vocab = set(corpus); VOCABSIZE = len(vocab)
  # map each unique word to an index for array indexing.
  vocab2ix = dict([(word, ix) for (ix, word) in enumerate(corpus)])
  # +ve and -ve sample vectors.
  # +ve vectors are random initialized, -ve vectors are zero initialized
  poss = np.rand((VOCABSIZE, DIMSIZE)); negs = np.zeros((VOCABSIZE, DIMSIZE))

  for wix in range(len(corpus)): # for every location in the corpus
    w = vocab2ix[corpus[wix]]  # find word at location,
    l = max(wix-WINDOWSIZE, 0); r = min(wix+WINDOWSIZE, len(corpus)-1) # take a window

    for w2ix in range(l, r+1): # word in window
        w2 = vocab2ix[corpus[w2ix]] # prallel.
        learn(l=poss[w], r=negs[w2], target=1.0)

    for _ in range(NNEGSAMPLES): # random words outside window.
        w2ix = random.randint(0, len(corpus)-1) # random word.
        w2 = vocab2ix[corpus[w2ix]]
      learn(l=poss[w], r=negs[w2], target=0.0) # perpendicular
  return { v: poss[vocab2ix[v]] for v in vocab }
```

What's word2vec?

```
def learn(l: np.array, r:np.array, target: float):
  """
  gradient descent on
  loss = (target - dot(l, r))^2 where l = larr[lix]; r = rarr[rix]
  """
  dot = np.dot(l, r); grad_loss = 2 * (target - out)
  #dloss/dl = 2 * (target - dot(l, r)) r
  #dloss/dr = 2 * (target - dot(l, r)) l
  lgrad =  EPSILON * grad_loss * r; rgrad =  EPSILON * grad_loss * l
  # l -= eps * dloss/dl; r -= eps * dloss/dr
  l += EPSILON * grad_loss * r;
  r += EPSILON * grad_loss * l

def train(corpus: list, DIMSIZE: int):
    for w2ix in range(l, r+1): # positive samples, parallell
        w2 = vocab2ix[corpus[w2ix]] # word in window
        learn(l=poss[w], r=negs[w2], target=1.0)
    for _ in range(NNEGSAMPLES): # negative samples: perpendicular.
        w2ix = random.randint(0, len(corpus)-1) # random word outside window.
        learn(l=poss[w], r=negs[w2], target=0.0) # perpendicular
```

Using `word2vec`

- Dot products capture similarity.

Using word2vec

- Dot products capture similarity.
- nope! *cosine similarity* captures similarity: $v \cdot w/|v||w|$.
- Vector space structure captures analogy: $\texttt{king} - \texttt{man} + \texttt{woman} = \texttt{queen}$. [Analogy]

Using word2vec

- Dot products capture similarity.
- nope! *cosine similarity* captures similarity: $v \cdot w / |v||w|$.
- Vector space structure captures analogy: $\mathtt{king} - \mathtt{man} + \mathtt{woman} = \mathtt{queen}$. [Analogy]
- nope! $\mathit{normalize}(\hat{\mathtt{king}} - \hat{\mathtt{man}} + \hat{\mathtt{woman}}) = \hat{\mathtt{queen}}$
- `word2vec`"vectors" are always normalized!

Using word2vec

- Dot products capture similarity.
- nope! *cosine similarity* captures similarity: $v \cdot w / |v||w|$.
- Vector space structure captures analogy: $\texttt{king} - \texttt{man} + \texttt{woman} = \texttt{queen}$. [Analogy]
- nope! *normalize*$(\hat{\texttt{king}} - \hat{\texttt{man}} + \hat{\texttt{woman}}) = \hat{\texttt{queen}}$
- word2vec"vectors" are always normalized!
- Cannot add, substract, scale them. So in what sense is the embedding "vectorial"?

Using `word2vec`

- Dot products capture similarity.
- nope! *cosine similarity* captures similarity: $v \cdot w / |v||w|$.
- Vector space structure captures analogy: $\texttt{king} - \texttt{man} + \texttt{woman} = \texttt{queen}$. [Analogy]
- nope! $\mathit{normalize}(\hat{\texttt{king}} - \hat{\texttt{man}} + \hat{\texttt{woman}}) = \hat{\texttt{queen}}$
- `word2vec`"vectors" are always normalized!
- Cannot add, substract, scale them. So in what sense is the embedding "vectorial"?
- In the sense that we have "vectors" — elements of the space $[-1, 1]^N$ with a normalization condition $(\sum_i x_i^2 = 1)$.

Using `word2vec`

- Dot products capture similarity.
- nope! *cosine similarity* captures similarity: $v \cdot w / |v||w|$.
- Vector space structure captures analogy: $\mathtt{king} - \mathtt{man} + \mathtt{woman} = \mathtt{queen}$. [Analogy]
- nope! $\mathit{normalize}(\hat{\mathtt{king}} - \hat{\mathtt{man}} + \hat{\mathtt{woman}}) = \hat{\mathtt{queen}}$
- `word2vec`"vectors" are always normalized!
- Cannot add, substract, scale them. So in what sense is the embedding "vectorial"?
- In the sense that we have "vectors" — elements of the space $[-1, 1]^N$ with a normalization condition $(\sum_i x_i^2 = 1)$.
- Can we ascribe a *different* meaning to these "vectors"?

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.
- A mathematical analogy: The *meaning* of an expression $\forall x \in \mathbb{Z}, x \leqslant 2$ is the *set* of possible values where the meaning holds true: $(-\infty, 2] = \{x \in \mathbb{Z} : x \leqslant 2\}$.

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.
- A mathematical analogy: The *meaning* of an expression $\forall x \in \mathbb{Z}, x \leqslant 2$ is the *set* of possible values where the meaning holds true: $(-\infty, 2] = \{x \in \mathbb{Z} : x \leqslant 2\}$.
- Meaning $\simeq$ subsets. Is `word2vec` subsets?

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.
- A mathematical analogy: The *meaning* of an expression $\forall x \in \mathbb{Z}, x \leqslant 2$ is the *set* of possible values where the meaning holds true: $(-\infty, 2] = \{x \in \mathbb{Z} : x \leqslant 2\}$.
- Meaning $\simeq$ subsets. Is `word2vec` subsets? Yes, *fuzzy sets*.

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.
- A mathematical analogy: The *meaning* of an expression $\forall x \in \mathbb{Z}, x \leqslant 2$ is the *set* of possible values where the meaning holds true: $(-\infty, 2] = \{x \in \mathbb{Z} : x \leqslant 2\}$.
- Meaning $\simeq$ subsets. Is `word2vec` subsets? Yes, *fuzzy sets*.
- Set: binary membership. $(1 \in_? \{1, 2\} = T$, $3 \notin_? \{1, 2\} = F)$.

Part I: What's a philosopher to do?

- Montague semantics: The *meaning* of a word is the *set* of possible worlds where the meaning holds true.
- A mathematical analogy: The *meaning* of an expression $\forall x \in \mathbb{Z}, x \leqslant 2$ is the *set* of possible values where the meaning holds true: $(-\infty, 2] = \{x \in \mathbb{Z} : x \leqslant 2\}$.
- Meaning $\simeq$ subsets. Is `word2vec` subsets? Yes, *fuzzy sets*.
- Set: binary membership. ($1 \in_? \{1, 2\} = T$, $3 \notin_? \{1, 2\} = F$).
- Fuzzy set: probabilistic membership. ($1 \in_{fuz} F = 0.1$, $2 \in_{fuz} F = 0.5$).

The hidden sets in `word2vec`

- Given the set of vectors, normalize the ith component of the vector across *all vectors*.

The hidden sets in word2vec

- Given the set of vectors, normalize the ith component of the vector across *all vectors*.
- $\texttt{fuzembed}_{\texttt{word}}[i] \equiv \texttt{vecembed}_{\texttt{word}}[i] / \sum_{w \in \texttt{CORPUS}} \texttt{vecembed}_{w}[i]$.

The hidden sets in `word2vec`

- Given the set of vectors, normalize the *i*th component of the vector across *all vectors*.
- $\mathtt{fuzembed}_{\mathtt{word}}[i] \equiv \mathtt{vecembed}_{\mathtt{word}}[i] / \sum_{w \in \mathtt{CORPUS}} \mathtt{vecembed}_{w}[i]$.
- Fuzzy set embedding from `word2vec` embeddings.

The hidden sets in `word2vec`

- Given the set of vectors, normalize the *i*th component of the vector across *all vectors*.
- $\texttt{fuzembed}_{\texttt{word}}[i] \equiv \texttt{vecembed}_{\texttt{word}}[i] / \sum_{w \in \texttt{CORPUS}} \texttt{vecembed}_{w}[i]$.
- Fuzzy set embedding from `word2vec` embeddings.

What does this buy us anyway? (Set operations)

$$(A \cap B)[i] \equiv A[i] \times B[i] \quad \text{(set intersection)}$$
$$(A \cup B)[i] \equiv A[i] + B[i] - A[i] \times B[i] \, \text{(set union)}$$
$$(A \sqcup B)[i] \equiv \max(1, \min(0, A[i] + B[i])) \, \text{(disjoint union)}$$
$$(\neg A)[i] \equiv 1 - A[i] \quad \text{(complement)}$$
$$(A \setminus B)[i] \equiv A[i] - \min(A[i], B[i]) \quad \text{(set difference)}$$
$$(A \subseteq B) \equiv \forall x \in \Omega : \mu_A(x) \leqslant \mu_B(x) \, \text{(set inclusion)}$$
$$|A| \equiv \sum_{i \in \Omega} \mu_A(i) \quad \text{(cardinality)}$$

What does this buy us anyway? (Entropy)

Fuzzy entropy is a measure of the uncertainty of the elements belonging to the set.

What does this buy us anyway? (Entropy)

Fuzzy entropy is a measure of the uncertainty of the elements belonging to the set.

$$
\begin{aligned}
H(A) &\equiv \sum_i H(X_i^A) \\
&\equiv \sum_i -p_i^A \ln p_i^A - (1-p_i^A)\ln\left(1-p_i^A\right) \\
&\equiv \sum_i -A[i]\ln A[i] - (1-A[i])\ln(1-A[i])
\end{aligned}
$$

and	the	in	one	which	to	however	*two*	for	*eight*	
this	of	of	in	the	*zero*	to	is	a	for	On the
as	and	only	a	also	*nine*	it	as	but	*s*	

left: Top 15 words with highest entropy with frequency $\geqslant 100$ (note that all of them are function words). On the right: Top 15 words with the highest frequency. The non-function words have been emphasized for comparison.

What does this buy us anyway? (KL divergence)

- K-L (Kullback Leibler) divergence is an asymmetric measure of similarity.
- Given data d which follows distribution P, the extra bits need to store it under the false assumption that the data d follows distribution Q is the K-L divergence between the distributions P and Q.
- Let P be the distribution that assigns 0.25 probability to a, b, c, d. Since all are equiprobable, we use 2 bits per character.
- Let Q be the distribution that assigns 0.5 probability to a, b and 0 probability to c, d. We use 1 bit to represent if we are storing a or b.
- If the real distribution is Q and we store data using P, then we really need only $\{a, b\}$, but we are trying to store $\{a, b, c, d\}$. P(false assumption) needs twice as many bits as Q(true distribution) to store the message c.
- If the real distribution is P and we store data using Q, then we really need $\{a, b, c, d\}$, but we *can only store* $\{a, b\}$. Q(false assumption) need *infinitely* more bits to store the message c than P (true distribution).

What does this buy us anyway? (KL divergence)

$$KL(S,T) \equiv \sum_i KL(X_i^S, X_i^T) = \sum_i p_i^S \log\left(p_i^S / p_i^T\right)$$

Example 1	$KL(ganges, delta)$	6.3105
	$KL(delta, ganges)$	6.3040
Example 2	$KL(north \cap korea, china)$	1.02923
	$KL(china, north \cap korea)$	10.60665

- K-L divergence shows the relation between two words.
- Can also consider phrases when composed using feature intersection as in the case of north korea.
- We demonstrate human annotator judgement of the distance between China and North Korea, where human annotators considered "North Korea" to be very similar to "China", while the reverse relationship was rated as significantly less strong ("China" is not very similar to "North Korea")

What does this buy us anyway? (Cross entropy)

$\hat{N}$	$\hat{M}$	$\hat{G}$	$\hat{N} \cap \hat{M}$	$\hat{N} \cap \hat{G}$
nobility	metal	bad	fusible	good
isotope	fusible	manners	unreactive	dharma
fujwara	ductility	happiness	metalloids	morals
feudal	with	evil	ductility	virtue
clan	alnico	excellent	heavy	righteous
$\vec{N}$	$\vec{M}$	$\vec{G}$	$\vec{N} + \vec{M}$	$\vec{N} + \vec{G}$
noblest	trivalent	bad	fusible	gracious
auctoritas	carbides	natured	metals	virtuous
abies	metallic	humoured	sulfides	believeth
eightfold	corrodes	selfless	finntroll	savages
vojt	alloying	gracious	rhodium	hedonist

- Polysemy of the word `noble`, in the context of the words `good` and `metal`.
- `noble` is represented by N, `metal` by M and `good` by G.
- We also provide the word2vec analogues of the same, under $\vec{N}$, $\vec{M}$, and $\vec{G}$.
- See that `word2vec` has no analogue for set-intersection. We use the closest possible analogue (addition), which performs worse semantically.

Take-aways

Conclusion

- `word2vec` is performant but poorly understood.
- We extract fuzzy set embeddings from `word2vec`, appeasing Montague!
- We ponder on the geometry of `word2vec`, and indicate potential extensions.
- TL;DR: Mathematical modelling (fuzzy sets) is useful to extend empirical results (`word2vec`)!
- `https://www.aclweb.org/anthology/2020.repl4nlp-1.4/`