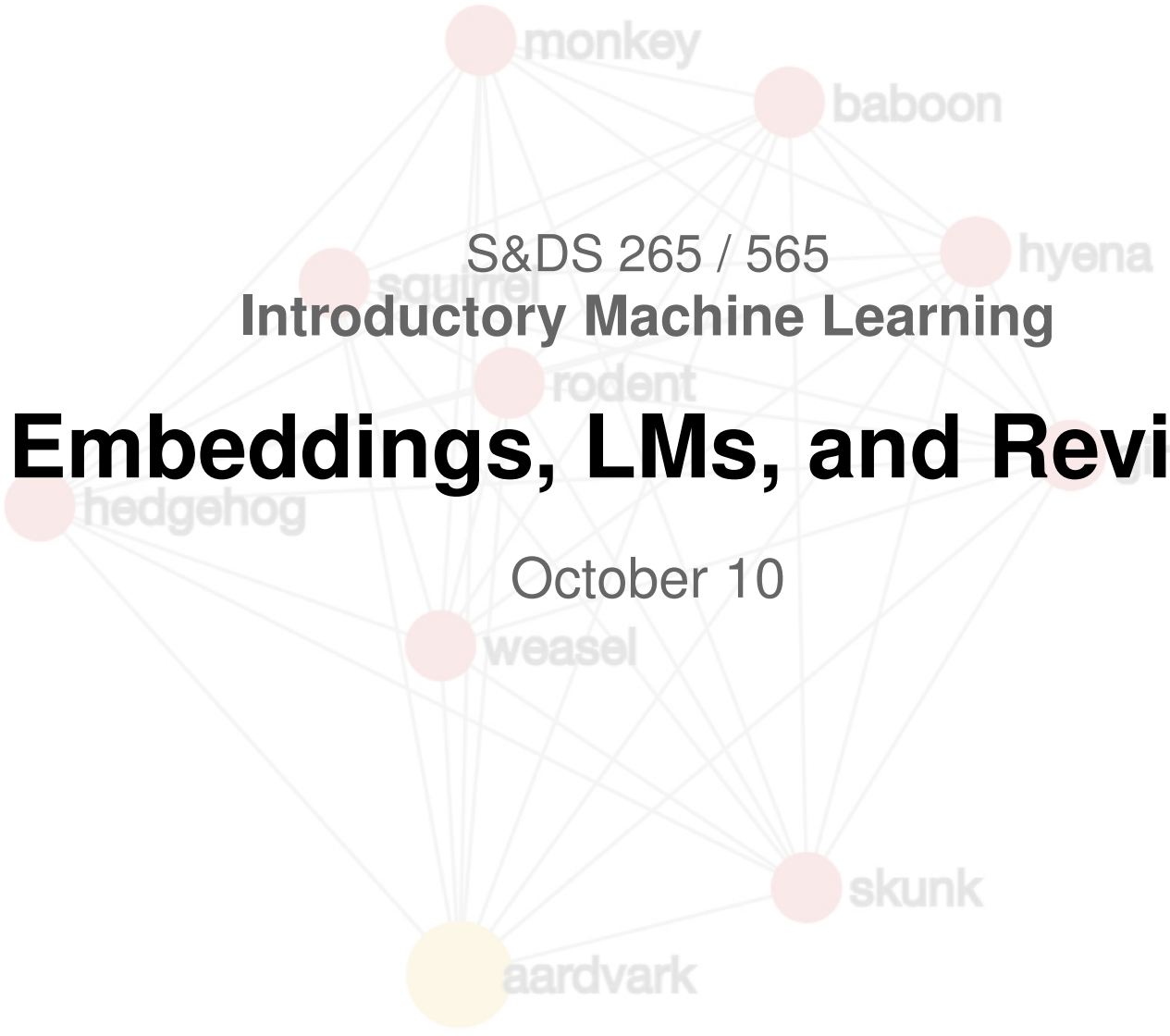


S&DS 265 / 565

Introductory Machine Learning

Embeddings, LMs, and Review

October 10

Yale

Plan for today

- Reminders
- Recap of word embeddings
- More on language modeling
- Review–AMA

Reminders

- Assn 3 is out; due October 24
- Quiz 3 posted today; open at 1pm; discriminitive vs. generative models, trees, bias/variance, SGD – good review for midterm!
- Midterm Tuesday, October 15, in class
- “Closed book, notes, computer...”
- $8\frac{1}{2} \times 11$ sheet of notes, handwritten double-sided
- Practice midterms posted on Canvas (with solutions)

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

- By the basic rules of conditional probability we can factor this as

$$p(w_1, \ldots, w_n) = p(w_1)p(w_2 \mid w_1) \ldots p(w_n \mid w_1, \ldots, w_{n-1})$$

Modern language models

Suppose a computer program assigns a "score" to possible next words v:

$$s(v;\ \overbrace{w_1, \ldots, w_n}^{\text{previous words}}\ )$$

v: possible next word

The score is given by

$$s(v; w_1, \ldots, w_n) = \beta_v^T \varphi(w_1, \ldots, w_n)$$

where $\varphi(w_1, \ldots, w_n)$ is an encoding of the context into a big vector, and β_v is an "embedding" of word v.

Modern language models

Can convert this to a language model by the “softmax” operation:

$$p(w \mid w_1, \ldots, w_n) = \frac{\exp(\beta_w^T \varphi(w_1, \ldots, w_n))}{\sum_{v \in V} \exp(\beta_v^T \varphi(w_1, \ldots, w_n))}$$

Modern language models

Can convert this to a language model by the "softmax" operation:

$$p(w \mid w_1, \ldots, w_n) = \frac{\exp(\beta_w^T \varphi(w_1, \ldots, w_n))}{\sum_{v \in V} \exp(\beta_v^T \varphi(w_1, \ldots, w_n))}$$

In ChatGPT, the function $s(v; w_{1:n})$ is learned on large amounts of text (unsupervised) using a type of deep neural network called a *transformer*.

Embeddings

Word embeddings can be learned using a simple case

$$s(v; w_1, \ldots, w_n) = \varphi(v)^T \varphi(w_n)$$

So, we just train the language model to predict the next word by mapping each word to a vector, and using the similarity between vectors.

Key intuition

- Similar words will appear with similar words
- Self-referential notion of similarity

Constructing embeddings

Language model is

$$p(w_2 \mid w_1) = \frac{\exp(\varphi(w_2)^T \varphi(w_1)}{\sum_w \exp(\varphi(w)^T \varphi(w_1))}.$$

Carry out stochastic gradient descent over the embedding vectors $\varphi \in \mathbb{R}^d$ (where $d \approx 50$–500 is chosen by hand)

This is what Mikolov et al. (2014, 2015) did at Google.

"Distributed representations of words," (2014) "Efficient representations of words in vector space" (2015)

Constructing embeddings

`word2vec`:

- Skip-gram: predict surrounding words from current word, rather than the next word.

"Distributed representations of words," (2014) "Efficient representations of words in vector space" (2015)

Constructing embeddings

`word2vec`:

- Skip-gram: predict surrounding words from current word, rather than the next word.
- This leads to a model of nearby words $p_{\text{near}}(w_2 \mid w_1)$.

"Distributed representations of words," (2014) "Efficient representations of words in vector space" (2015)

GloVe

Shortly after, a group at Stanford group introduced a variant called "GloVe"

- Based on a type of regression model
- More scalable with SGD

$$\mathcal{L}(\varphi) = \sum_{w_1, w_2} f(c_{w_1,w_2}) \left(\varphi(w_1)^T \varphi(w_2) - \log c_{w_1,w_2} \right)^2$$

where $c_{w,w'}$ are cooccurrence counts in a window (PMI)

Pennington et al., "GloVe: Global vectors for word representation," (2015)

Using PCA

A closely related approach is to use PCA of pointwise mutual information (PMI):

- Form $V \times V$ matrix of pointwise mutual information values

$$\log\left(\frac{p_{\text{near}}(w_1, w_2)}{p(w_1)p(w_2)}\right)$$

- Compute top k eigenvectors $\varphi_1, \ldots, \varphi_k$
- For each word w, define embedding as

$$\varphi(w) \equiv (\varphi_{1w}, \varphi_{2w}, \ldots, \varphi_{kw})^T$$

Analogies

Leads to vector representations of words with interesting properties.

For example, analogies:

`king` is to `man` as ? is to `woman`

Analogies

Leads to vector representations of words with interesting properties.

For example, analogies:

`king` is to `man` as ? is to `woman`

`Paris` is to `France` as ? is to `Germany`

Analogies

Leads to vector representations of words with interesting properties.

For example, analogies:

$$\texttt{king} \text{ is to } \texttt{man} \text{ as ? is to } \texttt{woman}$$

$$\texttt{Paris} \text{ is to } \texttt{France} \text{ as ? is to } \texttt{Germany}$$

$$\varphi(\texttt{king}) - \varphi(\texttt{man}) \overset{?}{\approx} \varphi(\texttt{queen}) - \varphi(\texttt{woman})$$

$$\widehat{w} = \arg\min_{w} \|\varphi(\texttt{king}) - \varphi(\texttt{man}) + \varphi(\texttt{woman}) - \varphi(w)\|^2$$

Does $\widehat{w} = \texttt{queen}$?

Learned Analogies

Table 8: *Examples of the word pair relationships, using the best word vectors from Table 4 (Skip-gram model trained on 783M words with 300 dimensionality).*

Relationship	Example 1	Example 2	Example 3
France - Paris	Italy: Rome	Japan: Tokyo	Florida: Tallahassee
big - bigger	small: larger	cold: colder	quick: quicker
Miami - Florida	Baltimore: Maryland	Dallas: Texas	Kona: Hawaii
Einstein - scientist	Messi: midfielder	Mozart: violinist	Picasso: painter
Sarkozy - France	Berlusconi: Italy	Merkel: Germany	Koizumi: Japan
copper - Cu	zinc: Zn	gold: Au	uranium: plutonium
Berlusconi - Silvio	Sarkozy: Nicolas	Putin: Medvedev	Obama: Barack
Microsoft - Windows	Google: Android	IBM: Linux	Apple: iPhone
Microsoft - Ballmer	Google: Yahoo	IBM: McNealy	Apple: Jobs
Japan - sushi	Germany: bratwurst	France: tapas	USA: pizza

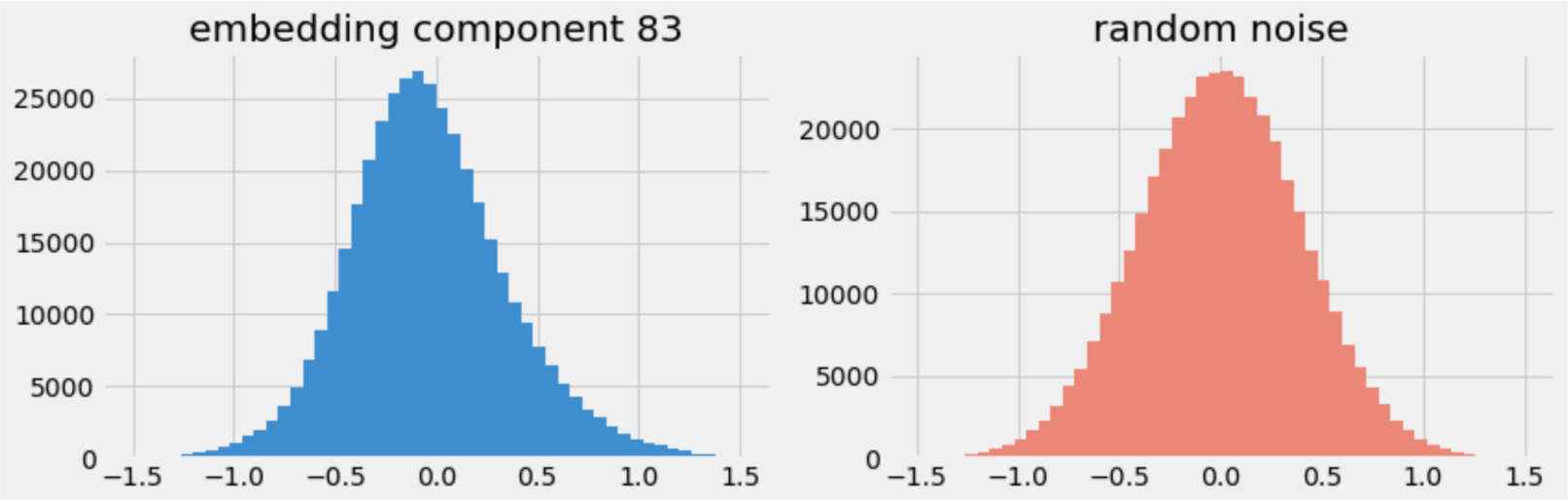
Mikolov et al., “Distributed representations of words,” (2014); “Efficient representations of words in vector space” (2015)

Evaluation Analogies

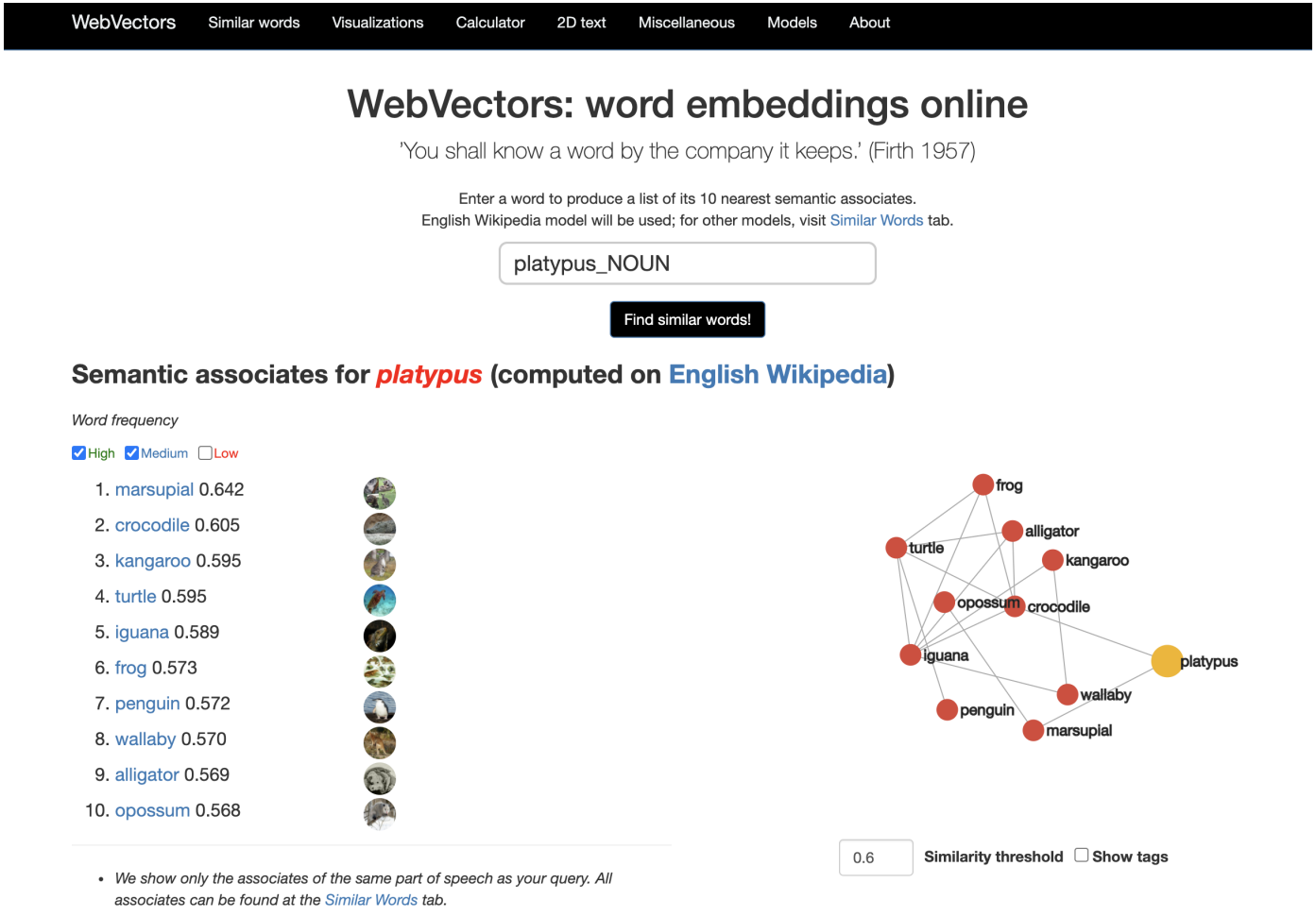
Type of relationship	Word Pair 1		Word Pair 2	
Common capital city	Athens	Greece	Oslo	Norway
All capital cities	Astana	Kazakhstan	Harare	Zimbabwe
Currency	Angola	kwanza	Iran	rial
City-in-state	Chicago	Illinois	Stockton	California
Man-Woman	brother	sister	grandson	granddaughter
Adjective to adverb	apparent	apparently	rapid	rapidly
Opposite	possibly	impossibly	ethical	unethical
Comparative	great	greater	tough	tougher
Superlative	easy	easiest	lucky	luckiest
Present Participle	think	thinking	read	reading
Nationality adjective	Switzerland	Swiss	Cambodia	Cambodian
Past tense	walking	walked	swimming	swam
Plural nouns	mouse	mice	dollar	dollars
Plural verbs	work	works	speak	speaks

Mikolov et al., “Distributed representations of words,” (2014); “Efficient representations of words in vector space” (2015)

Notebook


embedding component 83
25000
20000
15000
10000
5000
0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
random noise
20000
15000
10000
5000
0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5

Embedding / Visualization Examples



http://vectors.nlpl.eu/explore/embeddings/en/

Many uses

species2vec: A novel method for species representation

Boyan Angelov

doi: https://doi.org/10.1101/461996

Abstract

Word embeddings are omnipresent in Natural Language Processing (NLP) tasks. The same technology which defines words by their context can also define biological species. This study showcases this new method - species embedding (species2vec). By proximity sorting of 6761594 mammal observations from the whole world (2862 different species), we are able to create a training corpus for the skip-gram model. The resulting species embeddings are tested in an environmental classification task. The classifier performance confirms the utility of those embeddings in preserving the relationships between species, and also being representative of species consortia in an environment.

Visualisation

In [10]:
```
m = gensim.models.KeyedVectors.load_word2vec_format('reptilia.vec')
```

In [11]:
```
len(m.vocab)
```

Out[11]:
```
7397
```

In [15]:
```
m.most_similar(u'Alligator_mississippiensis')
```

Out[15]:
```
[(u'Sternotherus_bonevalleyensis', 0.8425856828689575),
 (u'Apalone_ferox', 0.8147842884063721),
 (u'Macrochelys_suwanniensis', 0.8063992261886597),
 (u'Deirochelys_reticularia', 0.7871163487434387),
 (u'Terrapene_putnami', 0.7841686010360718),
 (u'Chelydra_floridana', 0.7829421758651733),
 (u'Alligator_mefferdi', 0.7742743492126465),
 (u'Macrochelys_temminckii', 0.7682404518127441),
 (u'Trachemys_inflata', 0.7563525438308716),
 (u'Deirochelys_carri', 0.755811333656311)]
```

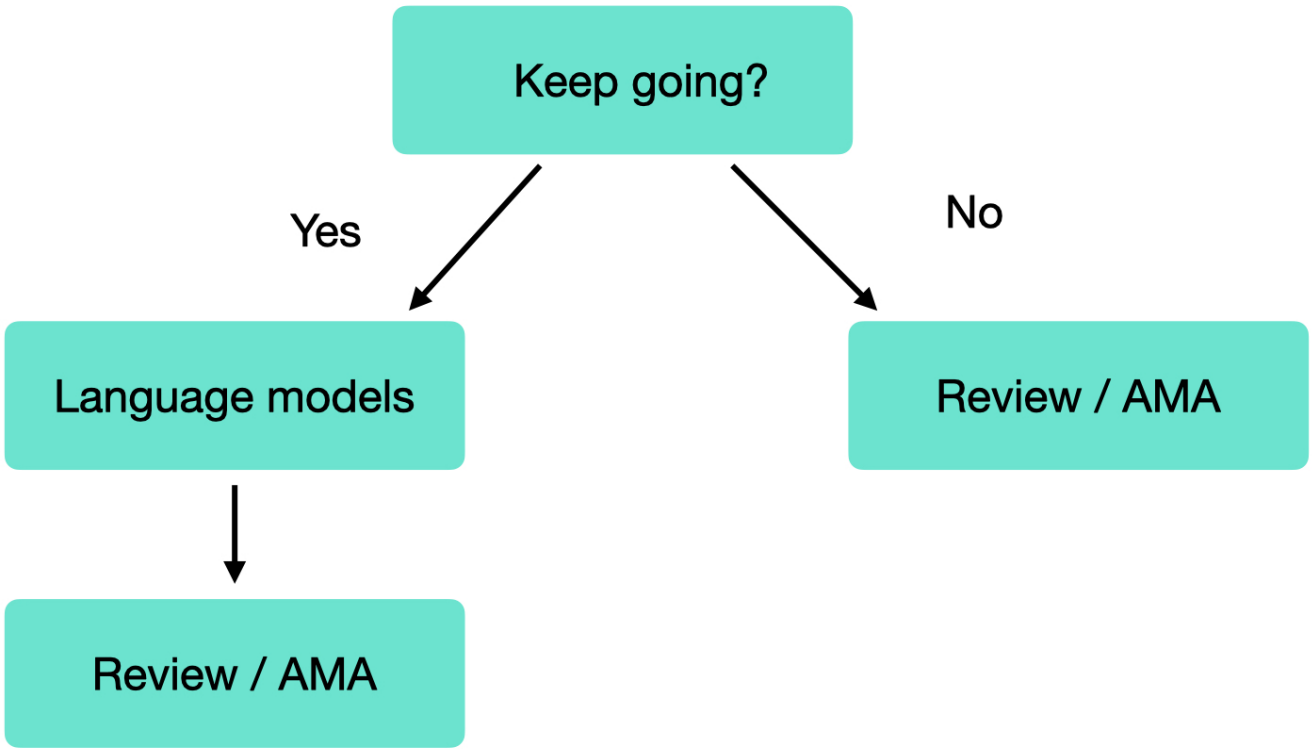
In [16]:
```
%matplotlib inline

def tsne_plot(model):
    "Creates and TSNE model and plots it"
    labels = []
```

Summary: Word embeddings

- Word embeddings are vector representations of words, learned from cooccurrence statistics
- The models can be built using language modeling (or regression or PCA)
- Surprising semantic relations are encoded in linear relations—for example, analogies
- Embeddings are the “ground floor” representations in ChatGPT

Decision Tree



Language models

- A language model is a way of *generating* any sequence of words

$$
\begin{aligned}
&P(\text{“the whole forest had been anesthetized”}) = \\
&P(\text{“the”}) \times P(\text{“whole”} \mid \text{“the”}) \\
&\times P(\text{“forest”} \mid \text{“the whole”}) \\
&\times P(\text{“had”} \mid \text{“the whole forest”}) \\
&\times P(\text{“been”} \mid \text{“the whole forest had”}) \\
&\times P(\text{“anesthetized”} \mid \text{“the whole forest had been”})
\end{aligned}
$$

Remixing Noon

Text generated from Channel Skin by Jeff Noon

"The whole forest had been anesthetised, her temples wired, her senses stimulated, her eyes were not on Eva, not on Eva, not on Eva, not on anybody in that same realm, the land of dreams and nightmares."

Viability: 0.000000326%

https://chatbotslife.com/notes-on-remixing-noon-generative-text-and-markov-chains-84ff4ec23937

Text generation

- Words generated one-by-one
- A word is chosen by sampling from a probability distribution
- Result is purely synthetic text—generative AI

Uses of language models

- Speech recognition
- Machine translation
- Text compression
- Texting
- Email completion
- Image captioning
- Mind reading from fMRI

Often built using Bayes' rule: $P(\text{signal} \mid \text{words}) \propto P(\text{words} \mid \text{signal}) \cdot P(\text{words})$

Dasher: LMs for assistive devices

Language models enable new modes of text input:

https://www.youtube.com/watch?v=quw_Kci4fUg

https://youtu.be/QxFEUk3J89Q?t=72

Dasher for text input, David Mackay, University of Cambridge

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

- By the basic rules of conditional probability we can factor this as

$$p(w_1, \ldots, w_n) = p(w_1)p(w_2 \mid w_1) \ldots p(w_n \mid w_1, \ldots, w_{n-1})$$

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

- By the basic rules of conditional probability we can factor this as

$$p(w_1, \ldots, w_n) = p(w_1)p(w_2 \mid w_1) \ldots p(w_n \mid w_1, \ldots, w_{n-1})$$

- The number of *histories* grows as V^{n-1}. Number of parameters in model grows as V^n, where V is number of words in vocabulary.

Language models

- A language model is a way of assigning a probability to any sequence of words (or string of text)

$$p(w_1, \ldots, w_n)$$

- By the basic rules of conditional probability we can factor this as

$$p(w_1, \ldots, w_n) = p(w_1)p(w_2 \mid w_1) \ldots p(w_n \mid w_1, \ldots, w_{n-1})$$

- The number of *histories* grows as V^{n-1}. Number of parameters in model grows as V^n, where V is number of words in vocabulary.
- What are some ways of reducing the number of parameters?

One approach: Grouping histories

- Let $\pi(w_1, \ldots, w_n)$ be the group assigned to the history

One approach: Grouping histories

- Let $\pi(w_1, \ldots, w_n)$ be the group assigned to the history
- Our model becomes

$$p(w_{n+1} \mid w_1, \ldots, w_n) = p(w_{n+1} \mid \pi(w_1, \ldots, w_n))$$

One approach: Grouping histories

- Let $\pi(w_1, \ldots, w_n)$ be the group assigned to the history
- Our model becomes

$$p(w_{n+1} \mid w_1, \ldots, w_n) = p(w_{n+1} \mid \pi(w_1, \ldots, w_n))$$

- Number of parameters: $O(V \cdot \text{number of groups})$

One approach: Grouping histories

- Let $\pi(w_1, \ldots, w_n)$ be the group assigned to the history
- Our model becomes

$$p(w_{n+1} \mid w_1, \ldots, w_n) = p(w_{n+1} \mid \pi(w_1, \ldots, w_n))$$

- Number of parameters: $O(V \cdot \text{number of groups})$
- What are some example groupings?

Grouping histories

- Unigrams: $\pi(w_1, \ldots, w_n) = \emptyset$.

The notation $O(\cdot)$ is called “Big Oh” and means “no greater than a constant times”, so that $O(f(n))$ means a sequence that is bounded by $C \cdot f(n)$ for large enough n.

Grouping histories

- Unigrams: $\pi(w_1, \ldots, w_n) = \emptyset$.
- Bigrams: $\pi(w_1, \ldots, w_n) = w_n$.

The notation $O(\cdot)$ is called "Big Oh" and means "no greater than a constant times", so that $O(f(n))$ means a sequence that is bounded by $C \cdot f(n)$ for large enough n.

Grouping histories

- Unigrams: $\pi(w_1, \ldots, w_n) = \emptyset$.
- Bigrams: $\pi(w_1, \ldots, w_n) = w_n$.
- Trigrams: $\pi(w_1, \ldots, w_n) = (w_{n-1}, w_n)$.

The notation $O(\cdot)$ is called "Big Oh" and means "no greater than a constant times", so that $O(f(n))$ means a sequence that is bounded by $C \cdot f(n)$ for large enough n.

Grouping histories

- Unigrams: $\pi(w_1, \ldots, w_n) = \emptyset$.
- Bigrams: $\pi(w_1, \ldots, w_n) = w_n$.
- Trigrams: $\pi(w_1, \ldots, w_n) = (w_{n-1}, w_n)$.
- Number of parameters grows as $O(V)$, $O(V^2)$, and $O(V^3)$, respectively.

The notation $O(\cdot)$ is called "Big Oh" and means "no greater than a constant times", so that $O(f(n))$ means a sequence that is bounded by $C \cdot f(n)$ for large enough n.

Estimating parameters

- The maximum likelihood estimate of a trigram model:

$$\widehat{p}(w_3 \mid w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

Estimating parameters

- The maximum likelihood estimate of a trigram model:

$$\widehat{p}(w_3 \mid w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

- What are some problems with this model?

Sparse data problem

The next group of slides presents one way of quantifying the problem of sparse data in language modeling

Half Earth

Aug 06, 2018 *by Foundation Staff* *0 Comment* *Google Earth, Half-Earth Project, Map of Life*

By Jeremy Malczyk, Michelle Duong, Ajay Ranipeta, Chris Heltne, Walter Jetz of Map of Life, Yale University, and the E.O. Wilson Biodiversity Foundation Half-Earth Project

This article originally in *Medium,* July 30, 2018

The Significance of Biodiversity

Narrow-billed Tody, Todus angustirostris. Photo by Julie Hart.

half-earth project

Learn how you can be part of bringing Half-Earth to life.

https://eowilsonfoundation.org/mapping-species-for-half-earth/

https://www.half-earthproject.org/

Specious species

- A naturalist (say, Edward O. Wilson) explores a region and observes animals (organisms).

Specious species

- A naturalist (say, Edward O. Wilson) explores a region and observes animals (organisms).
- He finds 100,000 animals and 5,000 species.

Specious species

- A naturalist (say, Edward O. Wilson) explores a region and observes animals (organisms).
- He finds 100,000 animals and 5,000 species.
- What is the chance I'll find a new species?

Specious species

- A naturalist (say, Edward O. Wilson) explores a region and observes animals (organisms).
- He finds 100,000 animals and 5,000 species.
- What is the chance I'll find a new species?
- What if Wilson observes 100 unique species?

Missing species: Good-Turing

- Wilson observes 100,000 animals and 5,000 species, 100 of them are unique (only one observation of that species).

---

https://en.wikipedia.org/wiki/Good-Turing_frequency_estimation

Missing species: Good-Turing

- Wilson observes 100,000 animals and 5,000 species, 100 of them are unique (only one observation of that species).
- Good-Turing: Estimate of the probability that the next animal is a new species? $\widehat{p}_{GT} = 100/100,000 = 10^{-3}$.

---

https://en.wikipedia.org/wiki/Good-Turing_frequency_estimation

Missing species: Good-Turing

- Wilson observes 100,000 animals and 5,000 species, 100 of them are unique (only one observation of that species).
- Good-Turing: Estimate of the probability that the next animal is a new species? $\widehat{p}_{GT} = 100/100,000 = 10^{-3}$.
- This is an estimate of the missing probability mass.

---

https://en.wikipedia.org/wiki/Good-Turing_frequency_estimation

Sparse data: Species $\approx$ words

- Suppose we have a corpus of 500 million words. I count trigrams and find that 50 million of them are unique.

Sparse data: Species $\approx$ words

- Suppose we have a corpus of 500 million words. I count trigrams and find that 50 million of them are unique.
- When I see a new trigram, 10% of the time it won't have been seen before. (This is a typical number.)

Sparse data: Species $\approx$ words

- Suppose we have a corpus of 500 million words. I count trigrams and find that 50 million of them are unique.
- When I see a new trigram, 10% of the time it won't have been seen before. (This is a typical number.)
- This means that the maximum likelihood estimate (MLE) is zero, and the probability that my model predicts the next word will be zero.

Sparse data: Species $\approx$ words

- Suppose we have a corpus of 500 million words. I count trigrams and find that 50 million of them are unique.
- When I see a new trigram, 10% of the time it won't have been seen before. (This is a typical number.)
- This means that the maximum likelihood estimate (MLE) is zero, and the probability that my model predicts the next word will be zero.
- The MLE is supported on the observed data. We need to spread out the probability over unseen events.

Estimating parameters

- The maximum likelihood estimate of a trigram model:

$$\widehat{p}(w_3 \mid w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

Estimating parameters

- The maximum likelihood estimate of a trigram model:

$$\widehat{p}(w_3 \mid w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3)}{\text{count}(w_1, w_2)}$$

- How can the model be strengthened?

Smoothing and Interpolation

Smoothing:

$$\widetilde{p}(w_3 \mid w_1, w_2) = \frac{\text{count}(w_1, w_2, w_3) + \varepsilon}{\text{count}(w_1, w_2) + V\varepsilon}$$

Smoothing and Interpolation

Linear interpolation:

$$p(w_3 \mid w_1, w_2) = \lambda_3 \widehat{p}(w_3 \mid w_1, w_2) + \lambda_2 \widehat{p}(w_3 \mid w_2) + \lambda_1 \widehat{p}(w_3)$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

This is a type of “mixture model”

Modern language models

The algorithm assigns a “score” to possible next words v:

$$s(v;\ \overbrace{w_1, \ldots, w_n}^{\text{previous words}}\ )$$

v: possible next word

The score is given by

$$s(v; w_1, \ldots, w_n) = \beta_v^T \varphi(w_1, \ldots, w_n)$$

where $\varphi(w_1, \ldots, w_n)$ is an encoding of the context into a big vector, and β_v is an “embedding” of word v.

Converted into a probability with softmax

1	Aug 29	Course overview		Thu: Course overview		
2	Sept 3, 5	Python and background concepts	Python elements Covid trends	Tue: Python elements Thu: Pandas and linear regression	Data8 Chapters 3, 4, 5	Quiz 1 Assn 1 out
3	Sept 10, 12	Linear regression and classification	Covid trends (revisited) Classification examples	Tue: Regression concepts Thu: Classification	ISL Sections 3.1, 3.2, 3.5 Notes on regression ISL Sections 4.3, 4.4 Notes on classification	
4	Sept 17, 19	Stochastic gradient descent	SGD examples	Tue: Classification (continued) Thu: Stochastic gradient descent	ISL Section 6.2.2 ISL Section 10.7.2	Assn 1 in Assn 2 out
5	Sept 24, 26	Bias and variance, cross-validation	Bias-variance tradeoff Covid trends (revisited) California housing	Tue: Bias and variance Thu: Cross-validation	ISL Section 2.2 ISL Section 5.1	Quiz 2
6	Oct 1, 3	Tree-based methods and principal components	Trees and forests Visualizing trees PCA examples	Tue: Trees (and Forests) Thu: Forests and PCA	ISL Sections 8.1, 8.2 ISL Section 12.2	Assn 2 in Assn 3 out
7	Oct 8, 10	PCA and dimension reduction	PCA revisited Used for dimension reduction Word	Tue: PCA and word embeddings Thu: Embeddings and review	ISL Section 12.2	Quiz 3

AMA