WIKIPEDIA

Language model

A **language model** is a probability distribution over sequences of words.[1] Given such a sequence of length m, a language model assigns a probability $P(w_1, \ldots, w_m)$ to the whole sequence. Language models generate probabilities by training on text corpora in one or many languages. Given that languages can be used to express an infinite variety of valid sentences (the property of digital infinity), language modelling faces the problem of assigning non-zero probabilities to linguistically valid sequences that may never be encountered in the training data. Several modelling approaches have been designed to surmount this problem, such as applying the Markov assumption or using neural architectures such as recurrent neural networks or transformers.

Language models are useful for a variety of problems in computational linguistics; from initial applications in speech recognition[2] to ensure nonsensical (i.e. low-probability) word sequences are not predicted, to wider use in machine translation[3] (e.g. scoring candidate translations), natural language generation (generating more human-like text), part-of-speech tagging, parsing,[3] Optical Character Recognition, handwriting recognition,[4] grammar induction,[5] information retrieval,[6][7] and other applications.

Language models are used in information retrieval in the query likelihood model. There, a separate language model is associated with each document in a collection. Documents are ranked based on the probability of the query Q in the document's language model M_d: $P(Q \mid M_d)$. Commonly, the unigram language model is used for this purpose.

Contents

Model types

Unigram

A unigram model can be treated as the combination of several one-state finite automata.[8] It assumes that the probabilities of tokens in a sequence are independent, e.g.:

$$P_{\text{uni}}(t_1 t_2 t_3) = P(t_1)P(t_2)P(t_3).$$

In this model, the probability of each word only depends on that word's own probability in the document, so we only have one-state finite automata as units. The automaton itself has a probability distribution over the entire vocabulary of the model, summing to 1. The following is an illustration of a unigram model of a document.

Terms	Probability in doc
a	0.1
world	0.2
likes	0.05
we	0.05
share	0.3
...	...

$$\sum_{\text{term in doc}} P(\text{term}) = 1$$

The probability generated for a specific query is calculated as

$$P(\text{query}) = \prod_{\text{term in query}} P(\text{term})$$

Different documents have unigram models, with different hit probabilities of words in it. The probability distributions from different documents are used to generate hit probabilities for each query. Documents can be ranked for a query according to the probabilities. Example of unigram models of two documents:

Terms	Probability in Doc1	Probability in Doc2
a	0.1	0.3
world	0.2	0.1
likes	0.05	0.03
we	0.05	0.02
share	0.3	0.2
...	...	...

In information retrieval contexts, unigram language models are often smoothed to avoid instances where *P*(term) = 0. A common approach is to generate a maximum-likelihood model for the entire collection and linearly interpolate the collection model with a maximum-likelihood model for each document to smooth the model.[9]

n-gram

In an *n*-gram model, the probability $P(w_1, \ldots, w_m)$ of observing the sentence $w_1, \ldots, w_m$ is approximated as

$$P(w_1, \ldots, w_m) = \prod_{i=1}^{m} P(w_i \mid w_1, \ldots, w_{i-1}) \approx \prod_{i=2}^{m} P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1})$$

It is assumed that the probability of observing the i^{th} word w_i in the context history of the preceding $i - 1$ words can be approximated by the probability of observing it in the shortened context history of the preceding $n - 1$ words (n^{th} order Markov property). To clarify, for *n=3* and *i=2* we have $P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) = P(w_2 \mid w_1)$.

The conditional probability can be calculated from *n*-gram model frequency counts:

$$P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{\mathrm{count}(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{\mathrm{count}(w_{i-(n-1)}, \ldots, w_{i-1})}$$

The terms **bigram** and **trigram** language models denote *n*-gram models with *n* = 2 and *n* = 3, respectively.[10]

Typically, the *n*-gram model probabilities are not derived directly from frequency counts, because models derived this way have severe problems when confronted with any *n*-grams that have not been explicitly seen before. Instead, some form of smoothing is necessary, assigning some of the total probability mass to unseen words or *n*-grams. Various methods are used, from simple "add-one" smoothing (assign a count of 1 to unseen *n*-grams, as an uninformative prior) to more sophisticated models, such as Good-Turing discounting or back-off models.

Bidirectional

Bidirectional representations condition on both pre- and post- context (e.g., words) in all layers.[11]

Example

In a bigram (*n* = 2) language model, the probability of the sentence *I saw the red house* is approximated as

$$P(\text{I, saw, the, red, house}) \approx P(\text{I} \mid \langle s \rangle) P(\text{saw} \mid \text{I}) P(\text{the} \mid \text{saw}) P(\text{red} \mid \text{the}) P(\text{house} \mid \text{red}) P(\langle /s \rangle \mid \text{house})$$

whereas in a trigram (*n* = 3) language model, the approximation is

$$P(\text{I, saw, the, red, house}) \approx P(\text{I} \mid \langle s \rangle, \langle s \rangle) P(\text{saw} \mid \langle s \rangle, I) P(\text{the} \mid \text{I, saw}) P(\text{red} \mid \text{saw, the}) P(\text{house} \mid \text{the, red}) P(\langle /s \rangle \mid \text{red, house})$$

Note that the context of the first *n* – 1 *n*-grams is filled with start-of-sentence markers, typically denoted <s>.

Additionally, without an end-of-sentence marker, the probability of an ungrammatical sequence **I saw the* would always be higher than that of the longer sentence *I saw the red house.*

Exponential

Maximum entropy language models encode the relationship between a word and the n-gram history using feature functions. The equation is

$$P(w_m \mid w_1, \ldots, w_{m-1}) = \frac{1}{Z(w_1, \ldots, w_{m-1})} \exp(a^T f(w_1, \ldots, w_m))$$

where $Z(w_1, \ldots, w_{m-1})$ is the partition function, a is the parameter vector, and $f(w_1, \ldots, w_m)$ is the feature function. In the simplest case, the feature function is just an indicator of the presence of a certain n-gram. It is helpful to use a prior on a or some form of regularization.

The log-bilinear model is another example of an exponential language model.

Neural network

Neural language models (or *continuous space language models*) use continuous representations or embeddings of words to make their predictions.[12] These models make use of Neural networks.

Continuous space embeddings help to alleviate the curse of dimensionality in language modeling: as language models are trained on larger and larger texts, the number of unique words (the vocabulary) increases.[a] The number of possible sequences of words increases exponentially with the size of the vocabulary, causing a data sparsity problem because of the exponentially many sequences. Thus, statistics are needed to properly estimate probabilities. Neural networks avoid this problem by representing words in a distributed way, as non-linear combinations of weights in a neural net.[13] An alternate description is that a neural net approximates the language function. The neural net architecture might be feed-forward or recurrent, and while the former is simpler the latter is more common.

Typically, neural net language models are constructed and trained as probabilistic classifiers that learn to predict a probability distribution

$$P(w_t \mid \mathrm{context}) \, \forall t \in V.$$

I.e., the network is trained to predict a probability distribution over the vocabulary, given some linguistic context. This is done using standard neural net training algorithms such as stochastic gradient descent with backpropagation.[13] The context might be a fixed-size window of previous words, so that the network predicts

$$P(w_t \mid w_{t-k}, \ldots, w_{t-1})$$

from a feature vector representing the previous k words.[13] Another option is to use "future" words as well as "past" words as features, so that the estimated probability is

$$P(w_t \mid w_{t-k}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+k}).$$

This is called a bag-of-words model. When the feature vectors for the words in the context are combined by a continuous operation, this model is referred to as the continuous bag-of-words architecture (CBOW).[14]

A third option that trains slower than the CBOW but performs slightly better is to invert the previous problem and make a neural network learn the context, given a word.[14] More formally, given a sequence of training words $w_1, w_2, w_3, \ldots, w_T$, one maximizes the average log-probability

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{-k \leq j \leq k, j \neq 0} \log P(w_{t+j} \mid w_t)$$

where k, the size of the training context, can be a function of the center word w_t. This is called a skip-gram language model.[15] Bag-of-words and skip-gram models are the basis of the word2vec program.[16]

Instead of using neural net language models to produce actual probabilities, it is common to instead use the distributed representation encoded in the networks' "hidden" layers as representations of words; each word is then mapped onto an n-dimensional real vector called the word embedding, where n is the size of the layer just before the output layer. The representations in skip-gram models have the distinct characteristic that they model semantic relations between words as linear combinations, capturing a form of compositionality. For example, in some such models, if v is the function that maps a word w to its n-d vector representation, then

$$v(\mathrm{king}) - v(\mathrm{male}) + v(\mathrm{female}) \approx v(\mathrm{queen})$$

where ≈ is made precise by stipulating that its right-hand side must be the nearest neighbor of the value of the left-hand side.[14][15]

Other

A positional language model[17] assesses the probability of given words occurring close to one another in a text, not necessarily immediately adjacent. Similarly, bag-of-concepts models[18] leverage the semantics associated with multi-word expressions such as *buy_christmas_present*, even when they are used in information-rich sentences like "today I bought a lot of very nice Christmas presents".

Despite the limited successes in using neural networks,[19] authors acknowledge the need for other techniques when modelling sign languages.

Evaluation and Benchmarks

Evaluation of the quality of language models is mostly done by comparison to human created sample benchmarks created from typical language-oriented tasks. Other, less established, quality tests examine the intrinsic character of a language model or compare two such models. Since language models are typically intended to be dynamic and to learn from data it sees, some proposed models investigate the rate of learning, e.g. through inspection of learning curves. [20]

Various data sets have been developed to use to evaluate language processing systems.[11] These include:

- Corpus of Linguistic Acceptability[21]
- GLUE benchmark[22]
- Microsoft Research Paraphrase Corpus[23]
- Multi-Genre Natural Language Inference
- Question Natural Language Inference
- Quora Question Pairs[24]
- Recognizing Textual Entailment[25]
- Semantic Textual Similarity Benchmark
- SQuAD question answering Test[26]
- Stanford Sentiment Treebank[27]
- Winograd NLI

Criticism

Although contemporary language models, such as GPT-2, can be shown to match human performance on some tasks, it is not clear they are plausible cognitive models. For instance, recurrent neural networks have been shown to learn patterns humans do not learn and fail to learn patterns that humans do learn.[28]

See also

- Statistical model
- Factored language model
- Cache language model
- Katz's back-off model
- Transformer
- BERT
- GPT-2
- GPT-3

Notes

a. See Heaps' law.

References

Citations

1. Jurafsky, Dan; Martin, James H. (2021). "N-gram Language Models". *Speech and Language Processing* (https://web.stanford.edu/~jurafsky/slp3/) (3rd ed.). Retrieved 24 May 2022.
2. Kuhn, Roland, and Renato De Mori. "A cache-based natural language model for speech recognition (https://www.researchgate.net/profile/Roland_Kuhn2/publication/3191800_Cache-based_natural_language_model_for_speech_recognition/links/004635184ee5b2c24f000000.pdf)." IEEE transactions on pattern analysis and machine intelligence 12.6 (1990): 570-583.
3. Andreas, Jacob, Andreas Vlachos, and Stephen Clark. "Semantic parsing as machine translation (https://www.aclweb.org/anthology/P13-2009)." Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). 2013.
4. Pham, Vu, et al. "Dropout improves recurrent neural networks for handwriting recognition (https://arxiv.org/abs/1312.4569)." 2014 14th International Conference on Frontiers in Handwriting Recognition. IEEE, 2014.
5. Htut, Phu Mon, Kyunghyun Cho, and Samuel R. Bowman. "Grammar induction with neural language models: An unusual replication (https://arxiv.org/pdf/1808.10000.pdf?source=post_page---------------------------)." arXiv preprint arXiv:1808.10000 (2018).
6. Ponte, Jay M.; Croft, W. Bruce (1998). *A language modeling approach to information retrieval*. Proceedings of the 21st ACM SIGIR Conference. Melbourne, Australia: ACM. pp. 275–281. doi:10.1145/290941.291008 (https://doi.org/10.1145%2F290941.291008).
7. Hiemstra, Djoerd (1998). *A linguistically motivated probabilistically model of information retrieval*. Proceedings of the 2nd European conference on Research and Advanced Technology for Digital Libraries. LNCS, Springer. pp. 569–584. doi:10.1007/3-540-49653-X_34 (https://doi.org/10.1007%2F3-540-49653-X_34).
8. Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze: An Introduction to Information Retrieval, pages 237–240. Cambridge University Press, 2009
9. Buttcher, Clarke, and Cormack. Information Retrieval: Implementing and Evaluating Search Engines. pg. 289–291. MIT Press.
10. Craig Trim, *What is Language Modeling?* (http://trimc-nlp.blogspot.com/2013/04/language-modeling.html), April 26th, 2013.
11. Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina (10 October 2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". arXiv:1810.04805 (https://arxiv.org/abs/1810.04805) [cs.CL (https://arxiv.org/archive/cs.CL)].
12. Karpathy, Andrej. "The Unreasonable Effectiveness of Recurrent Neural Networks" (https://karpathy.github.io/2015/05/21/rnn-effectiveness/).
13. Bengio, Yoshua (2008). "Neural net language models" (http://www.scholarpedia.org/article/Neural_net_language_models). *Scholarpedia*. Vol. 3. p. 3881. Bibcode:2008SchpJ...3.3881B (https://ui.adsabs.harvard.edu/abs/2008SchpJ...3.3881B). doi:10.4249/scholarpedia.3881 (https://doi.org/10.4249%2Fscholarpedia.3881).
14. Mikolov, Tomas; Chen, Kai; Corrado, Greg; Dean, Jeffrey (2013). "Efficient estimation of word representations in vector space". arXiv:1301.3781 (https://arxiv.org/abs/1301.3781) [cs.CL (https://arxiv.org/archive/cs.CL)].

15. Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado irst4=Greg S.; Dean, Jeff (2013). *Distributed Representations of Words and Phrases and their Compositionality* (http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf) (PDF). Advances in Neural Information Processing Systems. pp. 3111–3119.
16. Harris, Derrick (16 August 2013). "We're on the cusp of deep learning for the masses. You can thank Google later" (https://gigaom.com/2013/08/16/were-on-the-cusp-of-deep-learning-for-the-masses-you-can-thank-google-later/). *Gigaom*.
17. Lv, Yuanhua; Zhai, ChengXiang (2009). "Positional Language Models for Information Retrieval in" (http://times.cs.uiuc.edu/czhai/pub/sigir09-PLM.pdf) (PDF). *Proceedings*. 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR).
18. Cambria, Erik; Hussain, Amir (28 July 2012). *Sentic Computing: Techniques, Tools, and Applications* (https://books.google.com/books?id=NrtcLwEACAAJ). Springer Netherlands. ISBN 978-94-007-5069-2.
19. Mocialov, Boris; Hastie, Helen; Turner, Graham (August 2018). "Transfer Learning for British Sign Language Modelling" (https://www.aclweb.org/anthology/W18-3911/). *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*: 101–110. arXiv:2006.02144 (https://arxiv.org/abs/2006.02144). Retrieved 14 March 2020.
20. Karlgren, Jussi; Schutze, Hinrich (2015), "Evaluating Learning Language Representations", *International Conference of the Cross-Language Evaluation Forum*, Lecture Notes in Computer Science, Springer International Publishing, pp. 254–260, doi:10.1007/978-3-319-64206-2_8 (https://doi.org/10.1007%2F978-3-319-64206-2_8), ISBN 9783319642055
21. "The Corpus of Linguistic Acceptability (CoLA)" (https://nyu-mll.github.io/CoLA/). *nyu-mll.github.io*. Retrieved 25 February 2019.
22. "GLUE Benchmark" (https://gluebenchmark.com/). *gluebenchmark.com*. Retrieved 25 February 2019.
23. "Microsoft Research Paraphrase Corpus" (https://www.microsoft.com/en-us/download/details.aspx?id=52398). *Microsoft Download Center*. Retrieved 25 February 2019.
24. Aghaebrahimian, Ahmad (2017), "Quora Question Answer Dataset", *Text, Speech, and Dialogue*, Lecture Notes in Computer Science, vol. 10415, Springer International Publishing, pp. 66–73, doi:10.1007/978-3-319-64206-2_8 (https://doi.org/10.1007%2F978-3-319-64206-2_8), ISBN 9783319642055
25. Sammons, V.G.Vinod Vydiswaran, Dan Roth, Mark; Vydiswaran, V.G.; Roth, Dan. "Recognizing Textual Entailment" (http://l2r.cs.uiuc.edu/~danr/Teaching/CS546-12/TeChapter.pdf) (PDF). Retrieved 24 February 2019.
26. "The Stanford Question Answering Dataset" (https://rajpurkar.github.io/SQuAD-explorer/). *rajpurkar.github.io*. Retrieved 25 February 2019.
27. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank" (https://nlp.stanford.edu/sentiment/treebank.html). *nlp.stanford.edu*. Retrieved 25 February 2019.
28. Hornstein, Norbert; Lasnik, Howard; Patel-Grosz, Pritty; Yang, Charles (9 January 2018). *Syntactic Structures after 60 Years: The Impact of the Chomskyan Revolution in Linguistics* (https://books.google.com/books?id=XoxsDwAAQBAJ&dq=adger+%22goldilocks%22&pg=PA153). Walter de Gruyter GmbH & Co KG. ISBN 978-1-5015-0692-5.

Sources

- J M Ponte and W B Croft (1998). "A Language Modeling Approach to Information Retrieval". *Research and Development in Information Retrieval*. pp. 275–281. CiteSeerX 10.1.1.117.4237 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.117.4237).
- F Song and W B Croft (1999). "A General Language Model for Information Retrieval". *Research and Development in Information Retrieval*. pp. 279–280. CiteSeerX 10.1.1.21.6467 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.21.6467).
- Chen, Stanley; Joshua Goodman (1998). *An Empirical Study of Smoothing Techniques for Language Modeling* (Technical report). Harvard University. CiteSeerX 10.1.1.131.5458 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.131.5458).