Ordering-Sensitive and Semantic-Aware Topic Modeling Summary

Jashaina Thomas

Gaussian Mixture Neural Topic Model (GMNTM)

A major limitation of many topic models and their extensions is the bag-of-word assumption. This assumes that a document can be fully characterized by bag-of- word features. There has been little work on developing topic models where the order of words is taken into consideration.The Gaussian Mixture Neural Topic Model (GMNTM) considers both the ordering of words and the semantic meaning of sentences into the model. We represent each topic as a cluster of multi-dimensional vectors and embed the collection of texts into a collection of vectors generated by the Gaussian mixture model. Each word is affected not only by its topic, but also by the embedding vector of its surrounding words and the context. The Gaussian mixture components and the topic of documents, sentences and words can be learnt jointly. This model can learn better topics and more accurate word distributions for each topic, providing great performance in terms of perplexity, retrieval accuracy and classification accuracy in comparison to the best topic modeling approaches.

The GMNTM Model

This topic model is a Gaussian mixture model of vectors which encode words, sentences and documents. Each mixture component is associated with a specific topic. The word embeddings are learnt to optimize the predictability of a word using its surrounding words, with an important constraint that the vector representations are sampled from the Gaussian mixture which represents topics. In this model, words with similar semantics are more likely to be clustered into the same topic, and topics of sentences and documents are more accurately learned which potentially overcomes the weaknesses of the bag-of-word method.

We assume there are W different words in the vocabulary and there are D documents in corpus. For each word $w \in \{1, ..., W\}$ in vocabulary, there is an associated V-dimensional vector representation $vec(w) \in R^V$ for the word. Each document in corpus with index $d \in \{1, ..., D\}$also has a vector representation $vec(d) \in R^V$. If all the documents contain S sentences, then these sentences are indexed by $\in \{1, ..., S\}$. The sentence with index s is associated with a vector representation $vec(s) \in R^V$. There are T topics in the

GMNTM model, where T is designated by the user. Each topic corresponds to a Gaussian mixture component. The k-th topic is represented by V-dimensional Gaussian distribution $N(\mu_k, \sigma_k)$ with mixture weight $\pi_k \in R$, where $\mu_k \in R^V, \sum_k \in R^{VxV}$ and $\sum_{k=1}^{T} \pi_k = 1$. The parameters of the Gaussian mixture model are collectively represented by

$$\lambda = \{\pi_k, \mu_k, \sum_k\} \; k = 1, ..., T$$

Given the collection of parameters, we use

$$p(x|\lambda) = \sum_{i=1}^{T} \pi_i N(X|\mu_i, \sum_i)$$

to represent the probability distribution for sampling a vector x from the Gaussian mixture model.

For each word w in the vocabulary, we sample its topic z(w) from the multinomial distribution $\pi := (\pi_1, \pi_2, ..., \pi_T)$ and sample its vector representation $vec(w)$ from distribution $(\mu_{z(w)}, \sum_{z(w)})$. Equivalently, the vector vec(w) is sampled from the Gaussian mixture model parameterized by λ For each document d and each sentence s in the document, we sample their topics z(d), z(s) from distribution π and sample their vector representations, namely vec(d) and vec(s), also from the Gaussian mixture model. Let be the collection of latent vectors associated with all the words, sentences and documents in the corpus,

$$\Psi := \{vec(w)\} \cup \{vec(d)\} \cup \{vec(s)\}$$

For each word slot in the sentence, its word realization is generated according to the document's vector vec(d), the current sentence's vector vec(s) as well as at most m previous words in the same sentence. For the i-th location in the sentence, its word realization is w_i. The probability distribution of w_i is:

$$\begin{gathered} p(w_i = w|d, s, w_{i-m}X, ..., w) \\ \infty \exp(a_{doc}^w + a_{sen}^w + \sum_{t=1}^{m} a_t^w + b) \end{gathered}$$

a_{doc}, a_{sen} and a_t are influences from the document, the sentence and the previous word, respectively.

$$\begin{gathered} a_{doc}^w = \langle u_{doc}^w, vec(d) \rangle \\ a_{sen}^w = \langle wsen, vec(s) \rangle \\ a_t^w = \langle u_t^w, vec(w_{i-t}) \end{gathered}$$

u_{doc}^w, u_{sen}^w, $u_t^w \in R^V$ are parameters of the model, and they are shared across all slots in the corpus. U to representing this collection of vectors,

$$U := \{u_doc, u_sen\} \cup \{u_t | t \in 1, 2, ..., m\}$$

If we combine the equations above, the probability distribution of w_i is defined by a multi-class logistic model, where the features come from the vectors associated with the document, the sentence and the m previous words. Given the model parameters and the vectors for documents, sentences and words, we can infer the posterior probability distribution of topics. For a document d with vector representation $vec(d)$, the posterior distribution of its topic, namely $(z(d))$, is easy to calculate. For any $z \in 1, 2, ..., T$

$$q(z(d) = z) = \frac{\pi_z N(vec(d)|\mu_z, \sum_z)}{\sum_{k=1}^{T} \pi_k N(vec(d)|\mu_k, \sum_k)}$$

For each sentence s in the document d, the posterior distribution of its topic is

$$q(z(s) = z) = \frac{\pi_z N(vec(s)|\mu_z, \sum_z)}{\sum_{k=1}^{T} \pi_k N(vec(s)|\mu_k, \sum_k)}$$

For each word w in the vocabulary, the posterior distribution of its topic is similarly calculated as

$$q(z(w) = z) = \frac{\pi_z N(vec(w)|\mu_z, \sum_z)}{\sum_{k=1}^{T} \pi_k N(vec(w)|\mu_k, \sum_k)}$$

The probability of a word belonging to topic z proportional to the product of $q(z(w) = z), q(z(s) = z)$ and $q(z(d) = z)$, where w, s, and d are the word, the sentence and the document that this word slot associates with.

Estimating Model Parameters

The GMNTM model can be described as a probabilistic generative model. By estimating the model parameters, we learn the word representations that make one word predictable from its previous words and the context. Jointly, we learn the distribution of topics that words, sentences and documents belong to. We estimate the model parameters and by maximizing the likelihood of the generative model. This is done using the Expectation Maximization (EM) algorithm and stochastic gradient descent.

Step I: Estimating λ

In this stage, the latent vector of words, sentences and documents are fixed. We estimate the parameters of the Gaussian mixture model $\lambda = \{\pi_k, \mu_k, \sum_k\}$. This is a classical statistical estimation problem which can be solved by running the EM algorithm below.

EM Algorithm

• Inputs: A corpus containing D documents, S sentences, and a vocabulary containing W distinct words

• Initialize parameters

– Randomly initialize the vectors.

– Initialize parameters U with all-zero vectors.

– Initialize Gaussian mixture model parameters with the standard normal distribution

$N(0, diag(1))$.
• Repeat until converge
– Fixing parameters U and λ, run the EM algorithm to estimate the Gaussian mixture model parameter λ .
– Fixing the Gaussian mixture model λ, run stochastic gradient descent to maximize the log-likelihood of the model with respect to parameters U and Ψ.

Step II: estimating U and Ψ
When λ is known and fixed, we estimate the model parameters U and the latent vectors Ψ by maximizing the log-likelihood of the generative model. We iteratively sample a location in the corpus, and consider the log-likelihood of the observed word at this location. Let the word realization at location i be represented by w_i. The log-likelihood of this location is equal to

$$J_i(U, \Psi) = log(p(\Psi|\lambda)) + a_{doc}^{w_i} + a_{sen}^{w_i} + \sum_{t=1}^{m} a_t^{w_i} + b - log(\sum^{w} exp(a_{doc}^{w_i} + a_{sen}^{w_i} + \sum_{t=1}^{m} a_t^{w_i} + b))$$

$p(\Psi, \lambda)$ is the prior distribution of parameter Ψ in the Gaussian mixture model. The quantities a_{doc}^{w}, a_{sen}^{w} , and a_t^w are defined as in previous section.

The objective function $J_i(U, \Psi)$ involves all parameters in the collections (U, Ψ). Taking the computation efficiency into consideration, we only update the parameters associated with the word w_i. Concretely, we update

$$vec(w_i) \leftarrow vec(w_i) + \alpha \frac{\partial J_i(U,\Psi)}{\partial vec(w_i)}$$
$$u_t^{w_i} \leftarrow u_t^{w_i} + \alpha \frac{\partial J_i(U,\Psi)}{\partial u_t^{w_i}})$$

with α as the learning rate. We also update $vec(s)$, $vec(d)$ and $a_{doc}^{w_i}, a_{sen}^{w_i}$ using the same gradient step, as they are parameters associated with the current sentence and the current document. Once the gradient update is accomplished, we sample another location to continue the update. The process is finished when a sufficient number of updates performed and both U and Ψ converge to fixed values.

Reference; $Yang, M., Cui, T.$, and $Tu, W.$, 2015