ECE 684 Final Project

Document Classification

Clarissa Aché Cabello, Shufan Xia

Introduction:

In this project, we categorize research papers based on their abstract into one of the seven domains from Mechanical Engineering (MAE), Computer Science(CS), Electrical Engineering(ECE), Medical, Biochemistry, Civil, and Psychology.

The goal of this project is to implement a generative probabilistic model and a neural network model for this task. We used Latent Dirichlet Allocation (LDA) for the generative probabilistic approach. For the neural network approach, we applied Global Vectors for Word Representation (Glove) to represent words and implement a Bidirectional Long short-term memory (biLSTM) neural network. The two models are tested on prelabeled data from Web of Science (WoS) Dataset and synthetic data generated by the LDA probabilistic model. The performances of both approaches are compared.

Data Preprocessing

Due to computational limitations, only the top 2500 from the original WoS data are used. Raw text data are preprocessed before being fed to either the probabilistic or neural network. Preprocessing includes removing non-English and stop words, tokenizing and lemmatizing words. Preprocessing is carried out by the following steps.

1. Lowercase all characters and remove punctuations and numbers with regex`.`
2. Tokenize each abstract using NLTK.word_tokenizer(). This makes each document a list of tokens (words). The words “word” and “token” are interchangeable in this report.
3. Word included in NLTK English stopwords are removed by checking through each list of words (document) and keeping those not in the stopwords.
4. Lemmatization on each word using NLTK WordNetlLemmatizer(). This step groups together different forms of a word so they can be analyzed as a single item.
5. Remove most frequent words because academic papers tend to share a similar vocabulary. Words like 'use', 'model', 'system', 'study', 'method', 'propose', 'result', ‘find’, ‘develop’, ‘approach’ etc. are popular words used in academic papers across all domains. These words are not helpful to distinguish domains. We counted the frequency of each word after step 4 and removed the top 50 frequent words.

Generative probabilistic approach - LDA model for topic modeling

LDA model assumes each text can be represented by a bag of words, and each word is assigned with one topic. The most important parameters in LDA are α and β. α parameterizes the Dirtchelt distribution that determines the topic distribution of the document θ. θ is the multinomial distribution on the probability that a given word is assigned with a specific topic z. While β specifies the probability of each word in the vocabulary being generated if a given topic z. We utilize *gensim* package to implement LDA. We select the first 2500 documents in the WoS dataset, among which 2000 are used to extract α and β while the rest 500 is used as the validation dataset to evaluate the performance of this method. The method is broken down into 2 phases and 8 steps.

Phase 1: train an LDA model

1. Data preprocessing according to the discussion in the last section, but between steps 3 and 4 apply Gensim bigram to group two words that occur frequently. Because the premise of LDA is treating texts as bags of words, it would be better to consider two words that usually appear together as one token. `gensim.models.Phrases` provide a convenient way to detect and group phrases. `Min_count` is set to 50, and `threshold` to 100 for `gensim.models.Phrases()`.
2. Build the vocabulary from all training corpus and label each unique token from 1 to the size of the vocabulary. `gensim.corpora.dictionary.Dictionary` reads through all the documents in the training and takes out all the unique words from the corpus. It then generates a mapping between token to token_id, which we named as *word2id.*
3. Use the word to index mapping, *word2id,* to convert each tokenized document to a bag of words. Each document is converted to a list of tuples (token_id, token_count in a given document). The function `doc2bow()` under *gensim.corpora .dictionary.Dictionary* automatically makes this conversion.
4. Build the LDA model using *LdaModel* from *gensim.models* and train it on the corpus from step 3. We specify several parameters of this model as below:
   `lda_model = LdaModel(corpus, num_topics= 7 ,id2word = word2id, random_state=420, update_every=1, chunksize=100, passes=10, alpha='auto’)`
   Gensim infer the parameter *α* and β via Variational Bayes (VB) inference method. Understanding this method is beyond the scope of this project. Essentially it iterates over a batch of training documents and updates the model parameters with estimated values that optimize the probability of having the training documents until the distribution of model parameters converges. `Chunksize = 100` and `update_ever=1` say 100 documents are used in each batch of training, and the model parameters are updated after each batch of training. Smaller chunk size saves memory. `Passes = 10` means the is trained on the entire corpus 10 times in total. alpha stands for a priori belief on topic per document distribution. Since we have no prior knowledge of this distribution, it is set to “auto”. Finally, `random_state` determines the random seed for reproducibility.
5. Extract *α* and *β* from the trained LDA model by `lda_model.alpha` and `lda_model.get_topics()`.
6. Evaluate the performance on the LDA model and infer domain based on *β*. The performance of the model is quantified by perplexity score and coherence score. The result of the LDA model on the training corpus can be visualized using *pyLDAvis* package. It plots inter-topic distances and lists word frequency per topic. *β* outputted from the *gensim* LDA model is topics labeled by number and the word distribution per topic. To map each topic number back to each academic domain, we guess the domain based on word distribution per topic. We also consult the visualization from *pyLDvis* to make this inference. The number to domain mapping is saved a dictionary object.

Phase 2: use the LDA model after steps 1-6 to categorize any new document.

7. Any new document is cleaned up following the steps in Data Preprocessing and step1 above. Once it is cleaned, it is passed to the LDA model above. *gensim*

estimates the weight of each 7 topics for any given document using VB inference method.

8. Categorize the document by the topic number that has the largest topic weight. Topic numbers are mapped to the corresponding domain with the dictionary from step 6.

LDA is a generative probabilistic model. With α and β from step 4, we generated 2500 documents by the same process in homework 5, so the details won't be explained here. The length of the document is sampled from Poisson(λ) where λ is the median of token counts in each document after data preprocessing, 87 in our case. All 8 steps above are applied on the 2500 generated synthetic data. The performance of this method is discussed in the last section of the document.

Neural network approach - Glove and bi-LSTM

The LSTM neural network is a particular type of RNN that preserves the context of previously seen inputs by "saving" them in a hidden state. Using a bidirectional LSTM to run each abstract in two directions (forward and backward) has an advantage against the unidirectional LSTM because running the document backward allows the model to preserve context from both the beginning of the document and the end at the same time. Additionally, before training the model, the words in the input documents are embedded into vectors of dimension = 100 using *GloVe*, a pre-trained word-vector representation method. Similar to word2vec, *GloVe* embedding uses the entire corpus to understand and create the word representations. Given the vocabulary of the corpus is >20K words, using one-hot encoded vectors to represent each word would have made the model computationally inefficient. The complete workflow of the neural network is designed as below:

1. *Glove* is downloaded from *gensim* package. After data pre-processing, each token is vectorized with *Glove*. And these word vectors of dimension 100 are the inputs to our neural network.
2. The neural network consists of first a bi-LSTM and a linear layer. The bi-LSTM network uses the `torch.nn.LSTM` class with an input dimension of 100 from *GloVe* embedding and output hidden states of dimension 50. `bidirectional` is set to True. After the bidirectional LSTM, the hidden states of only the last time step, forward and backward, are concatenated and passed to a linear layer. The input dimension to the linear layer is, thus, 50*2 = 100, and the output dimension is the number of topics, 7. Before the final activation function, the model also makes zero (randomly) 20% of the input tensor using `torch.nn.dropout(0.2)`, reducing the sensitivity to specific weights of individual neurons, which ultimately helps prevent overfitting. Finally, a Softmax activation converts and normalizes a vector of 7 numbers into probabilities of each topic.
3. This neural network model uses the Cross-Entropy loss function and the Adam optimizer for training on 1000 iterations.
4. Each document is classified to the topic with the highest probability after Softmax.

Comparing the Performance of LDA and bi-LSTM methods:

The LDA method obtained a 44% accuracy overall for the 2000 training documents while 35% for 500 testing documents from the WoS dataset. The estimated distributions of α and β are listed in Appendix Fig 1. By examining β, we notice that the word-per-topic

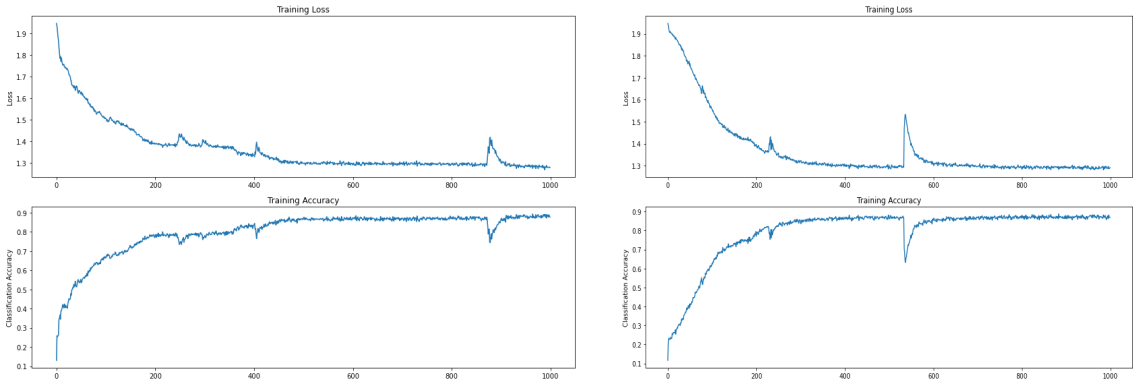
distribution is easily distinguishable for “CS”, “ECE”,” Psychology” and “Biochemistry”, but not for “Civil”, “MAE”, and Medical. For example, it is easy to associate "adolescent", "social” and “disorder” with Psychology; “signal”, “circuit”, and “controller” with ECE. On the other hand, it is not obvious that whether “fluid” and “wind” point to “Civil” or “MAE”, similar for Biochemistry and Medical. The percentage of correct categorization for each domain (Table 1 below) agrees with this finding: “Psychology” has the highest accuracy followed by “CS” and “Biochemistry”. The accuracy for “Civil” and “MAE” are both low. “Biochemistry” has higher accuracy while 0 for “Medical” because it is likely that most documents in “Medical” are categorized as “Biochemistry”. The overlaps between topics are also reflected in the topic inter-distance plot from *pyLDAvis* (Appendix Fig 4). “CS” and “ECE” are expected to have some overlaps, so do “Civil” and “MAE”. “Psychology” and “Biochemistry” are both distinct from the other topics. The bubble of “Medical” is expected to be close to the bubble of “Biochemistry”, but not in our result. This possibly implies the bubble labeled as “Biochemistry” includes documents from “Medical”, and the bubble labeled as “Medical” is, in fact, for some other latent topic.

The overall accuracy for the LDA method on the synthetic data was 75% and 73% for 2000 training documents and 500 testing documents respectively. Both are higher than the accuracy scores for the WoS dataset. This result is as expected because all the unseen documents in the 500 synthetic testing data follow the same topic-per-document and word-by-topic distribution of the 2000 synthetic training data. Again, the accuracies for “Civil” and “MAE”, “Biochemistry” and “Medical” are relatively lower because each pair of topics has overlaps. This conclusion is shown by the overlaps between the bubbles of “Civil” and “MAE”, “Biochemistry” and “Medical” in the topic inter-distance plot in Appendix Fig 5.

LDA method				Glove + bi-LSTM Neural Network			
WoS dataset		Synthetic data		WoS dataset		Synthetic data	
Domain	Accuracy	Domain	Accuracy	Domain	Accuracy	Domain	Accuracy
CS	0.77	CS	0.79	CS	0.6	CS	0.13
Civil	0.06	Civil	0.38	Civil	0.45	Civil	0.13
ECE	0.37	ECE	0.78	ECE	0.63	ECE	0.25
MAE	0	MAE	0.42	MAE	0.46	MAE	0.13
Medical	0	Medical	0.13	Medical	0.68	Medical	0.28
Psychology	0.91	Psychology	0.89	Psychology	0.53	Psychology	0.15
Biochemistry	0.64	Biochemistry	0.23	Biochemistry	0.49	Biochemistry	0.11

Table 1: Accuracy by domain for both methods on WoS dataset and the synthetic data.

The LSTM method produced a better result for the WoS dataset. The overall test accuracy for 500 documents was 57%, with partial accuracy for each domain ranging from 45% to 68% shown in detail in Table 1. However, on the synthetic data generated in the previous step, the accuracy is 19.4%. In Figures 2 (a) and (b), the training loss and accuracy are shown for the LSTM models using both real and synthetic data.



(a) (b)

Fig 2. Training progress for 2000 documents (a) from the WoS dataset and (b) from the synthetic data generated with our LDA method. The plots on the first row show the decrease in Cross Entropy Loss over training iteration from 0 to 1000, and the plots on the second row show the increase in the training accuracy. The Y-axis for each of the four plots is from 0.0 to 1.0.

The difference in performance between the “real” data from WoS and the synthetic can be explained by the additional variation that the LDA model adds in “guessing” the topics which, as explained in Step 6, were not perfect matches to the 7 original domains; that is in addition to the variation the generative probabilistic model adds in producing the synthetic data.

The LSTM neural network method offers a few advantages over the probabilistic approach. One is that it preserves context by keeping the order of words in the document, which is often an important aspect in deciding the paper domain given that most scientific research papers use similar words. The bidirectional LSTM model also offers an advantage in the embeddings used. GloVe embeddings hold more information about each word than the vectors used in the LDA model, which only offer information about the word’s position in the vocabulary.

Conclusions and Limitations

For both models, an important part of the performance is attributed to the preprocessing steps. The tokenization and lemmatization of words do reduce the morphological variation of words which allows the model to identify words in different syntactic or even orthographic forms as the same element. Additionally, by removing the 50 most common words in the corpus, both models become better at identifying nuances between different documents, given these very common words are expected to be uncorrelated with any topics.

The ability of the bidirectional LSTM to preserve context makes it a more accurate model in identifying the domains of the research paper abstracts on real data. However, on synthetic data, the LDA model, because the data was generated with the same parameters the model estimates about the data, is better at predicting the document’s domains.

As mentioned previously, LDA converges to groups of words that are more common in some latent topics, but does to provide the actual academic domain labels. The mapping between topic numbers to the actual academic domains, which can be subjective, had to be done manually by associating the most frequent words to domains. Lastly, it is plausible that one paper belongs to multiple domains or be interdisciplinary. In general, assigning a domain to a research paper is not an exact process which in reality depends largely on the institution

where paper and research were carried out. This means the word distribution of a paper does not always trully represent its domain.

As to computational costs, the LDA method ran faster than the Neural Network method significantly. It took 30-60 s for the *gensim* LDA model to converge, but 2 hours for training our bi-LSTM RNN in *pytorch*. One reason is that LDA is trained on lists of tuples, while our bi-LSTM RNN is trained on a 3D tensor of dimension (number of training documents, the maximum document length, *GloVe* embedding dimension), (2000, 292, 100) in our case. In addition, breaking the whole corpus into smaller chunks for training the LDA model saves some memory, as opposed to training on the whole corpus for our bi-LSTM RNN.

Reference

Blei, David M, Andrew Y Ng, and Michael I Jordan. 2003. “Latent Dirichlet Allocation.” *Journal of Machine Learning Research* 3 (Jan): 993–1022. https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

Matthew D. Hoffman, David M. Blei, and Francis Bach. 2010. Online learning for Latent Dirichlet Allocation. In <i>Proceedings of the 23rd International Conference on Neural Information Processing Systems - Volume 1 *NIPS'10*. Curran Associates Inc., Red Hook, NY, USA, 856–864.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. [pdf] [bib]

Prabhakaran, S.(March 26, 2018) *Topic Modeling with Gensim (Python)* https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/#20topicdistributionacrossdocuments

Tufts, C. (2018, January). *The Little Book of LDA.* Retrieved from https://LDAbook.com/index.html

Pai, A. (January 28, 2020) *Build Your First Text Classification model using PyTorch.* https://www.analyticsvidhya.com/blog/2020/01/first-text-classification-in-pytorch/

NS, A. *Multiclass Text Classification using LSTM in Pytorch- Predicting item ratings based on customer reviews.* https://towardsdatascience.com/multiclass-text-classification-using-lstm-in-pytorch-eac56baed8df

Dataset:

Kowsari, Kamran; Brown, Donald; Heidarysafa, Mojtaba ; Jafari Meimandi, Kiana ; Gerber, Matthew; Barnes, Laura (2018), “Web of Science Dataset”, Mendeley Data, V6, doi: 10.17632/9rw3vkcfy4.6 https://data.mendeley.com/datasets/9rw3vkcfy4/6

Appendix:

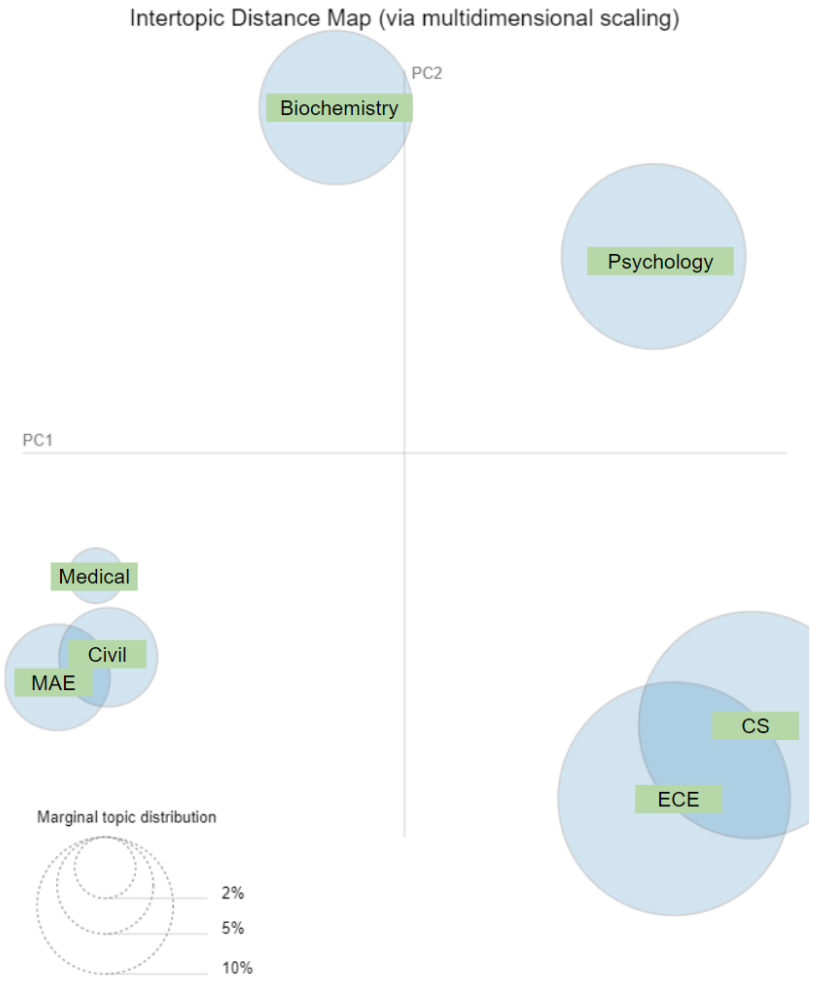


Fig 4. Intertopic distance plot from the LDA model on the WoS dataset.

α = [0.07, 0.92, 0.28, 0.88, 0.44, 0.17, 0.20]
β = [
(**"Medical"**, 0.009*"iron" + 0.008*"risk_factors" + 0.007*"fibrosis" + 0.006*"surgical" + 0.005*"cable"+ …),
(**"ECE"**, '0.005*"signal" + 0.005*"range" + 0.005*"reduce" + 0.005*"frequency" + 0.004*"stress"+ …),
(**"Civil"**, '0.011*"joint" + 0.009*"engine" + 0.007*"protocol" + 0.007*"emerge" + 0.007*"procedure"+ …),
(**"CS"**, '0.005*"work" + 0.005*"network" + 0.005*"problem" + 0.005*"research" + 0.004*"information"+ …),
(**"Psychology"**, '0.008*"adolescent" + 0.007*"group" + 0.006*"treatment" + 0.005*"social" + 0.004*"support" + …),
(**"MAE"**, '0.014*"flow" + 0.012*"water" + 0.009*"temperature" + 0.009*"heat" + 0.008*"wind"+ …),
(**"Biochemistry"**, '0.014*"expression" + 0.012*"gene" + 0.009*"protein" + 0.006*"activity" + 0.006*"human"+ …)
]

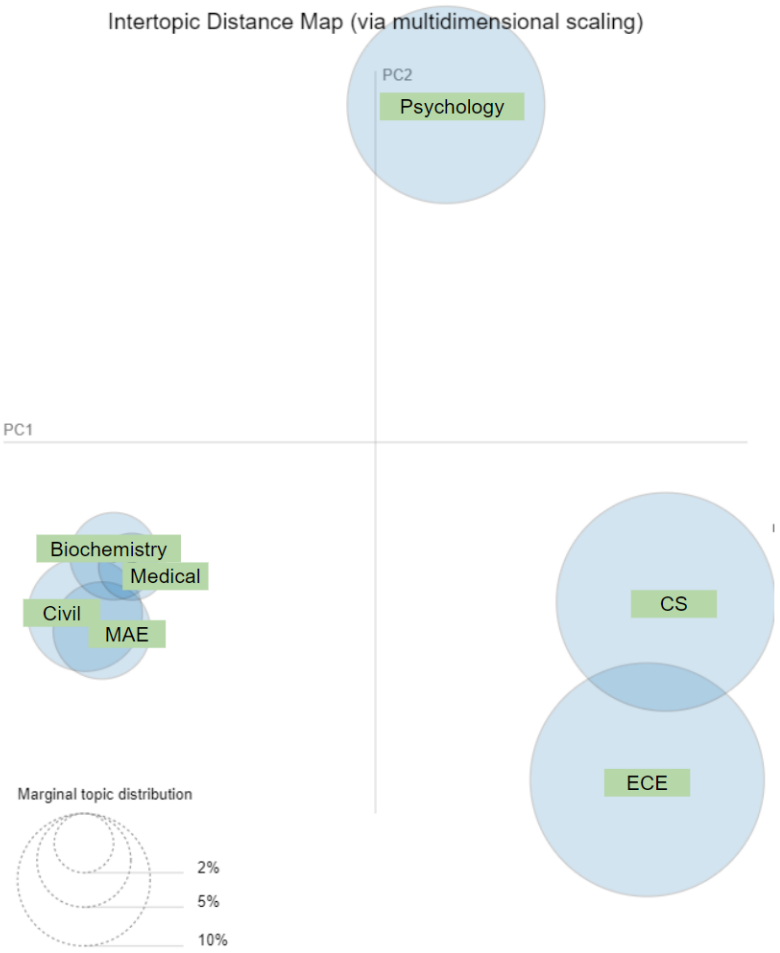


Fig 5. Intertopic distance plot from the LDA model on the synthetic data.

α =[0.44, 0.35, 2.51, 0.46, 0.81, 2.89, 2.17]

β = [

(**"MAE"**, '0.017*"joint" + 0.014*"engine" + 0.010*"emerge" + 0.009*"procedure" + 0.008*"capture"+ …),

(**"Medical"**, '0.008*"mitochondrial"+0.008*"soft"+0.005*"acs" + 0.005*"bioprosthetic" + 0.005*"risk_factors"+...),

(**"CS"**, '0.006*"problem" + 0.006*"work" + 0.006*"network" + 0.005*"information" + 0.004*"technology"+...),

(**"Biochemistry"**, '0.012*"protein" + 0.012*"gene" + 0.010*"expression" + 0.007*"cancer" + 0.006*"inhibitor" +...),

(**"Civil"**, '0.010*"flow" + 0.009*"water" + 0.008*"temperature" + 0.007*"heat" + 0.006*"wind" + …),

(**"ECE**", '0.006*"signal" + 0.005*"reduce" + 0.005*"response" + 0.005*"obtain" + 0.005*"frequency" + …'),

(**"Psychology"**, '0.006*"group" + 0.005*"treatment" + 0.005*"adolescent" + 0.005*"report" + 0.005*"associate" + …)

]