GuidedLDA Documentation

Release 1.0

Vikash Singh (vi3k6i5)

Oct 05, 2017

Contents:

`GuidedLDA` OR `SeededLDA` implements latent Dirichlet allocation (LDA) using collapsed Gibbs sampling. `GuidedLDA` can be guided by setting some seed words per topic. Which will make the topics converge in that direction.

You can read more about guidedlda in the documentation.

CHAPTER 1

Installation

```
pip install guidedlda
```

If pip install doesn't work then try the next step.

```
https://github.com/vi3k6i5/GuidedLDA
cd GuidedLDA
sh build_dist.sh
python setup.py sdist
pip install -e .
```

If that also doesn't work please raise issue with details of OS version, python version, architecture etc and I will try my best to fix it ASAP.

CHAPTER 2

Getting started

`guidedlda.GuidedLDA` implements latent Dirichlet allocation (LDA). The interface follows conventions found in scikit-learn.

Example Code.

The following demonstrates how to inspect a model of a subset of the Reuters news dataset. The input below, `X`, is a document-term matrix (sparse matrices are accepted).

```
>>> import numpy as np
>>> import guidedlda

>>> X = guidedlda.datasets.load_data(guidedlda.datasets.NYT)
>>> vocab = guidedlda.datasets.load_vocab(guidedlda.datasets.NYT)
>>> word2id = dict((v, idx) for idx, v in enumerate(vocab))

>>> X.shape
(8447, 3012)

>>> X.sum()
1221626
>>> # Normal LDA without seeding
>>> model = guidedlda.GuidedLDA(n_topics=5, n_iter=100, random_state=7, refresh=20)
>>> model.fit(X)
INFO:guidedlda:n_documents: 8447
INFO:guidedlda:vocab_size: 3012
INFO:guidedlda:n_words: 1221626
INFO:guidedlda:n_topics: 5
INFO:guidedlda:n_iter: 100
WARNING:guidedlda:all zero column in document-term matrix found
INFO:guidedlda:<0> log likelihood: -11489265
INFO:guidedlda:<20> log likelihood: -9844667
INFO:guidedlda:<40> log likelihood: -9694223
INFO:guidedlda:<60> log likelihood: -9642506
INFO:guidedlda:<80> log likelihood: -9617962
INFO:guidedlda:<99> log likelihood: -9604031
```

```
>>> topic_word = model.topic_word_
>>> n_top_words = 8
>>> for i, topic_dist in enumerate(topic_word):
>>>     topic_words = np.array(vocab)[np.argsort(topic_dist)][:-(n_top_words+1):-1]
>>>     print('Topic {}: {}'.format(i, ' '.join(topic_words)))
Topic 0: company percent market business plan pay price increase
Topic 1: game play team win player season second start
Topic 2: life child write man school woman father family
Topic 3: place open small house music turn large play
Topic 4: official state government political states issue leader case

>>> # Guided LDA with seed topics.
>>> seed_topic_list = [['game', 'team', 'win', 'player', 'season', 'second', 'victory
↪'],
>>>                    ['percent', 'company', 'market', 'price', 'sell', 'business',
↪'stock', 'share'],
>>>                    ['music', 'write', 'art', 'book', 'world', 'film'],
>>>                    ['political', 'government', 'leader', 'official', 'state',
↪'country', 'american','case', 'law', 'police', 'charge', 'officer', 'kill', 'arrest
↪', 'lawyer']]

>>> model = guidedlda.GuidedLDA(n_topics=5, n_iter=100, random_state=7, refresh=20)

>>> seed_topics = {}
>>> for t_id, st in enumerate(seed_topic_list):
>>>     for word in st:
>>>         seed_topics[word2id[word]] = t_id

>>> model.fit(X, seed_topics=seed_topics, seed_confidence=0.15)
INFO:guidedlda:n_documents: 8447
INFO:guidedlda:vocab_size: 3012
INFO:guidedlda:n_words: 1221626
INFO:guidedlda:n_topics: 5
INFO:guidedlda:n_iter: 100
WARNING:guidedlda:all zero column in document-term matrix found
INFO:guidedlda:<0> log likelihood: -11486362
INFO:guidedlda:<20> log likelihood: -9767277
INFO:guidedlda:<40> log likelihood: -9663718
INFO:guidedlda:<60> log likelihood: -9624150
INFO:guidedlda:<80> log likelihood: -9601684
INFO:guidedlda:<99> log likelihood: -9587803


>>> n_top_words = 10
>>> topic_word = model.topic_word_
>>> for i, topic_dist in enumerate(topic_word):
>>>     topic_words = np.array(vocab)[np.argsort(topic_dist)][:-(n_top_words+1):-1]
>>>     print('Topic {}: {}'.format(i, ' '.join(topic_words)))
Topic 0: game play team win season player second point start victory
Topic 1: company percent market price business sell executive pay plan sale
Topic 2: play life man music place write turn woman old book
Topic 3: official government state political leader states issue case member country
Topic 4: school child city program problem student state study family group
```

The document-topic distributions should be retrived as: `doc_topic = model.transform(X)`.

```
>>> doc_topic = model.transform(X)
>>> for i in range(9):
>>>     print("top topic: {} Document: {}".format(doc_topic[i].argmax(),
                                                  ', '.join(np.
↪array(vocab)[list(reversed(X[i,:].argsort()))[0:5]])))
top topic: 4 Document: plant, increase, food, increasingly, animal
top topic: 3 Document: explain, life, country, citizen, nation
top topic: 2 Document: thing, solve, problem, machine, carry
top topic: 2 Document: company, authority, opera, artistic, director
top topic: 3 Document: partner, lawyer, attorney, client, indict
top topic: 2 Document: roll, place, soon, treat, rating
top topic: 3 Document: city, drug, program, commission, report
top topic: 1 Document: company, comic, series, case, executive
top topic: 3 Document: son, scene, charge, episode, attack
```

Save the model for production or for running later:

```
>>> from six.moves import cPickle as pickle
>>> # Uncomment next step if you want to lighten the model object
>>> # This step will delete some matrices inside the model.
>>> # you will be able to use model.transform(X) the same way as earlier.
>>> # you wont be able to use model.fit_transform(X_new)
>>> # model.purge_extra_matrices()
>>> with open('guidedlda_model.pickle', 'wb') as file_handle:
>>>     pickle.dump(model, file_handle)
>>> # load the model for prediction
>>> with open('guidedlda_model.pickle', 'rb') as file_handle:
>>>     model = pickle.load(file_handle)
>>> doc_topic = model.transform(X)
```

CHAPTER 3

Requirements

Python 2.7 or Python 3.3+ is required. The following packages are required

- numpy
- pbr

CHAPTER 4

Caveat

`guidedlda` aims for Guiding LDA. More often then not the topics we get from a LDA model are not to our setisfaction. GuidedLDA can give the topics a nudge in the direction we want it to converge. We have production trained it for half a million documents (We have a big machine). We have run predictions on millions and manually checked topics for thousands (we are sattisfied with the results).

If you are working with a very large corpus you may wish to use more sophisticated topic models such as those implemented in hca and MALLET. hca is written entirely in C and MALLET is written in Java. Unlike `guidedlda`, hca can use more than one processor at a time. Both MALLET and hca implement topic models known to be more robust than standard latent Dirichlet allocation.

CHAPTER 5

Notes

Latent Dirichlet allocation is described in Blei et al. (2003) and Pritchard et al. (2000). Inference using collapsed Gibbs sampling is described in Griffiths and Steyvers (2004). And Guided LDA is described in Jagadeesh Jagarlamudi, Hal Daume III and Raghavendra Udupa (2012)

CHAPTER 6

Important links

- Documentation: http://guidedlda.readthedocs.org
- Source code: https://github.com/vi3k6i5/guidedlda/
- Issue tracker: https://github.com/vi3k6i5/guidedlda/issues

CHAPTER 7

Other implementations

- scikit-learn's LatentDirichletAllocation (uses online variational inference)
- gensim (uses online variational inference)

CHAPTER 8

Credits

I would like to thank creators of LDA project. I used the code from that LDA project as base to implement GuidedLDA on top of it.

Thanks to : Allen Riddell and Tim Hopper. :)

CHAPTER 9

License

guidedlda is licensed under Version 2.0 of the Mozilla Public License.

CHAPTER 10

Indices and tables

- genindex
- modindex
- search