Advanced Machine Learning: Assignment 1

Seung-Hoon Na

December 5, 2020

1 Explainable methods for deep learning

Deep learning has been recognized as a breakthrough approach that leads to significant improvements over the existing machine learning methods, in the area of computer vision and natural language processing, etc [1, 2, 3]. Despite its success, one of the important limiations in deploying the deep learning method is that the prediction made by neural models lacks *interpretability* and *explainability* that help human to determine whether he/she trusts the prediction results. Thus, providing explainability to decisions or predictions made by deep learning models becomes one of the most important issues, which refers to as *explainable deep learning*.

Explainable deep learning is roughly categorized to 1) *visualization methods*, 2) *model distillation*, and 3) *intrinsic methods*, as reviewed in [4]. First, visualization methods are to obtain *saliency maps* that indicate the extent how importantly input features affect outputs resulting from a model. Without training new separate models or changing original models, the major part of visualization methods is to define a mapping function between saliency of input features and outputs. Second, model distillation is to train a new simpler and interpretable model, which is separately designed from an original model. To make a new simple model faithful to the outputs of original model, the loss function needs to capture local behaviors of the original one at least in an approximate manner. Usually, model distillation methods are model-agnostic, in the sense that the distillation method can be applied to any type of model, without requiring special consideration of the given model. Third, intrinsic methods are to directly derive or extract explanation from the inference process inherent in the model architecture. Intrinsic methods assume that the model architecture itself contains the reasoning path that is related to the explanation. For example, the attention mechanisms used in transformer architecture have widely been used as evidences for relevance (or saliency) scores of input features influencing the model's outputs based on the attentive representation. Once a group of intrinsic models is identified and known, intrinsic models can be more preferred over yet-non-interpretable models (e.g. Transformer vs. LSTM) if the explainability should be fulfilled. Intrinsic methods may be helpful to improve the capability of model distillation, since a class of intepretable and simple models used by model distillation can be expanded by new models equipped with successful intrinsic methods.

In this project, we will study and implement three explanation methods– two visualization methods and one method of model distillation – **Layer-wise relevance propagation** (LRP) [5], **Grad-CAM** [6], and **LIME** [7].

The goal of this project is summarized as follows:

1. Completely review LRP, Grad-GAM, and LIME with the detailed derivations of them.

2. Implement LRP, Grad-CAM, and LIME using python

3. Empirical comparison of LRP, Grad-GAM, and LIME on various datasets.

2 Layer-wise relevance propagation (LRP)

Read the paper of LRP [5] and its applications to text classification [8] and image classification [9] and write a report that addresses the following problems.

2.1 Model

1. Summarize the key idea and formulation of LRP of [5]

2. Summarize ϵ-LRP (Eq. (58)) and Algorithm 2 of [5]

3. Why ϵ is necessary in Eq. (58) of [5]?

3 Layer-wise relevance propagation (LRP) for text classification

3.1 Text classification: Implementation using neural bag-of-words model for text classification using pytorch

Read [10] and implement neural Bag-of-Words (NBOW) model based on Eq. (1)-(2) of [10] using `pytorch`.

For pretrained word embedding models, use GloVe.
`https://nlp.stanford.edu/projects/glove/`
For evaluation, use the Large Movie Review Dataset v1.0.
`https://ai.stanford.edu/ amaas/data/sentiment/aclImdb_v1.tar.gz`
Evaluate the implemented 1-D ConvNet on the large movie review dataset.

3.2 Text classification: Implementation using 1-D ConvNet for text classification using pytorch

Implement the 1-D ConvNet for text classification, using `pytorch` in your own codes, by referring to section 2 of [11], where multiple window sizes, where windows sizes are $L = \{1, 2, 3\}$

3.3 Layer-wise relevance propagation (LRP) for text classification

1. Summarize the formula of 1-D ConvNet for text classification (as in Section 2 of [11]), when a single window size l is used.

2. Based on Section 3.4 of [8], derive the word-level relevance score in Eq. (5) of [8], and rewrite the equation using vector notations of [11]) (a single window size is used)

3. What is the difference b/w Eq. (6) and Eq. (7) of [8]? (word-level extraction vs. element-wise extraction).

4. Derive Eq. (5)-Eq. (7) of [8], when using 1-D Convnet using *multiple* window sizes L, as in the work of [11].

5. When $L = \{1, 2, 3\}$, unigram, bigram, and trigrams are used for text classification. Derive n-gram relevance score Eq. (5) ($n \in \{1, 2, 3\}$), document vectors Eq. (6)-(7) of [8] by extending of Eq. (5) of [8].

Evaluate the implemented 1-D ConvNet on the large movie review dataset, comparing to NBOW model.

3.4 Text classification: Implementation of ϵ-LRP using python

Implement ϵ-LRP in your own python codes for 1-D ConvNet with *multiple* window sizes $L = \{1, 2, 3\}$.

Check ϵ-LRP to highlight important words according their relevance scores.

3.5 Text classification: Per-instance KNN-based extrinsic validation method

Read Section 4.2 of [8] and summary the procedure of the *extrinsic validation* method of evaluating explanation models using document summary vectors.

3.6 LRP for Text classification: Experiments on large movie review dataset

Perform experiments of applying ϵ-LRP under 1-D ConvNet with *multiple window sizes* $L = \{1, 2, 3\}$ implemented in Section 3.2. The main experiments are given as follows:

1. Select top K words or n-grams and check how these words affect the classification accuracy under NBOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).

2. Compute two variants of document summary vectors Eq. (6)-(7) of [8] and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

For each experiment, write additional codes and scripts, prepare a README file that describe how to execute your codes or scripts with clear details.

More details for experiments are given as follows:

3.6.1 ϵ-LRP on large movie review dataset

1. Select top-K words (unigrams) according to ϵ-LRP, feed only the *selected words* to NBOW models and report the classification accuracy. Also, remove the *selected words* in a given sentence and feed the *remaining words* to NBOW models and report the classification accuracy. Perform these experiments using the various numbers of K from $\{1, \cdots, 10\}$

2. Select top-K n-grams (a set of unigram, bigrams, and trigrams) according to your derivation of ϵ-LRP, feed only the *selected n-grams* to NBOW models and report the classification accuracy. Also, remove the *selected n-grams* in a given sentence and feed the *remaining words* to NBOW models and report the classification accuracy. Perform these experiments using the various numbers of K from $\{1, \cdots, 10\}$

3. Using your derivation of the document summary vectors Eq. (6)-(7) of [8] for the 1-D ConvNet model based on multiple window sizes $\{1, 2, 3\}$, perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8] and compare the results of using the word-level extraction and the element-wise extraction for computing the document summary vector.

4 Layer-wise relevance propagation (LRP) for image classification

4.1 Image classification: Implementation of a bag-of-words model for image classification using python

In the bag-of-words representation in computer vision, an image is represented as a *bag-of-visual-words* (BoW), similar to the bag-of-words document representation [12, 13].

Read [12] and summarize the BoW model, addressing the following.

1. What are the keypoints? How to obtain them?

2. What are *visual words*? How to generate them? How to determine its vocabulary size?

3. Discuss various ways of obtaining *visual-word vectors* from the BOW representation.

Read the BoW generation part in Section VI-A.(2) of [12] with details.

Implement the BOW model based on the SIFT descriptor to detect the keypoints, the K-means clustering to generate visual vocabulary, and the SVM to conduct the classification.

Evaluate the BOW model on `Pascal VOC 2009` dataset.
`https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/D3GIPK`

4.2 Image classification: Implementation of convolutional networks using pytorch

Summarize the formula of convolutional networks for image classification (as in Section 2.2 of [9]).

Implement the convolutional network for image classification of Section 2.2 of [9] (or a similar architecture) using `pytorch` in your own codes.

Evaluate the convolutional model on `Pascal VOC 2009` dataset, comparing to the BOW model.

4.3 Layer-wise relevance propagation (LRP) for image classification

Read [9] and summarize ϵ-LRP for image classification

1. Derive ϵ-LRP formual for convolutional networks for image classification (as in Section 2.2 of [9]).
2. Based on the BOW representation using visual words, derive the word-level relevance score in Eq. (5) of [8] (Based on Section 3.4 of [8])
3. Based on the BOW representation using visual words, derive the formula of *image* summary vectors Eq. (6)-Eq. (7) of [8].

4.4 Image classification: Per-instance KNN-based extrinsic validation method

Similar to the text classification case, summary the procedure of the *extrinsic validation* method of evaluating explanation models *using* image summary vectors.

4.5 LRP for image classification: Experiments on Pascal VOC 2009 dataset

Perform experiments of applying ϵ-LRP under the implemented convolutional network. Similar to the text classification, the main experiments are given as follows:

1. Select top K *visual* words and check how these visual words affect the classification accuracy under the BOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).
2. Compute two variants of *image* summary vectors Eq. (6)-(7) of [8] and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

Also, for each experiment, you need to write additional codes and scripts, prepare a README file that describe how to execute your codes or scripts with clear details.

More details for experiments are given as follows:

4.5.1 ϵ-LRP on Pascal VOC 2009 dataset

1. Select top-K visual words (unigrams) according to ϵ-LRP, feed only the *selected visual words* to the *BOW* models and report the classification accuracy. Also, remove the *selected visual words* in a given sentence and feed the *remaining visual words* to *BOW* models and report the classification accuracy. Perform these experiments using the various numbers of K from $\{1, \cdots, 10\}$
2. Using your derivation of the *image* summary vectors Eq. (6)-(7) of [8] for the convolutional model of Section 2.2 of [9], perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8] and compare the results of using the word-level extraction and the element-wise extraction for computing the document summary vector.

5 Grad-CAM

5.1 Grad-CAM for deep learning

Read [6] and summarize the key idea and the formulation of Grad-CAM.

5.2 Grad-CAM for text classification: Implementation

Implement Grad-CAM for visualizing your 1-D convnet model implemented in Section 3.2

5.3 Grad-CAM for text classification: Document summary vectors

Derive the formulation based on **Grad-CAM** to compute document vectors Eq. (6)-(7) of [8]

5.4 Grad-CAM for text classification: Experiments

Similar to Section 3.6, perform experiments of applying **Grad-CAM** under the 1-D ConvNet of Section 3.2. The main experiments are given as follows:

1. Select top K words or n-grams using **Grad-CAM** and check how these words affect the classification accuracy under NBOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).

2. Compute two variants of document summary vectors Eq. (6)-(7) of [8] using **Grad-CAM** and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

Other details are the same as Section 3.6.

5.5 Grad-CAM for image classification: Implementation

Implement Grad-CAM for visualizing your convolutional networks implemented in Section 4.2.

5.6 Grad-CAM for image classification: Image summary vectors

Derive the formulation based on **Grad-CAM** to obtain image summary vectors Eq. (6)-(7) of [8].

5.7 Grad-CAM for image classification: Experiments

Similar to Section 4.5, perform experiments of applying **Grad-CAM** under the **convolutional networks** of Section 4.2. The main experiments are given as follows:

1. Select top K *visual* words or n-grams using **Grad-CAM** and check how these words affect the classification accuracy under NBOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).

2. Compute two variants of *image* summary vectors Eq. (6)-(7) of [8] using **Grad-CAM** and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

Other details are the same as Section 3.6.

6 LIME

6.1 LIME for deep learning

Read [7] and summarize the key idea, the main procedure, and Algorithm 1 of [7].

6.2 LIME for text classification: Implementation using python

Implement LIME for explaining your 1-D convnet model implemented in Section 3.2, addressing the following.

1. How to sample local examples given a specific instance?

2. How many samples are used as local examples for approximating the decision boundary?

3. Summarize K-LASSO used in Algorithm 1 of [7].

For reference, you can check Section 4.3 of [14].

6.3 LIME for text classification: Document summary vectors

Derive the formulation based on the results of **LIME** to compute document vectors Eq. (6)-(7) of [8]

6.4 LIME for text classification: Experiments

Similar to Section 3.6, perform experiments of applying **LIME** under the 1-D ConvNet of Section 3.2. Again, the main experiments are given as follows:

1. Select top K words or n-grams using **LIME** and check how these words affect the classification accuracy under NBOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).

2. Compute two variants of document summary vectors Eq. (6)-(7) of [8] and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

Likewise, other details are the same as Section 3.6.

6.5 LIME for image classification: Implementation

Implement **LIME** for explaining your convolutional networks implemented in Section 4.2.

6.6 LIME for image classification: Image summary vectors

Derive the formulation based on **LIME** to obtain image summary vectors Eq. (6)-(7) of [8].

6.7 LIME for image classification: Experiments

Similar to Section 4.5, perform experiments of applying **Grad-CAM** under the **convolutional networks** of Section 4.2. Again, the main experiments are given as follows:

1. Select top K *visual* words or n-grams using **LIME** and check how these words affect the classification accuracy under NBOW models, after 1) only keeping the selected words or ngrams and 2) removing those selected ones (leave-K-out).

2. Compute two variants of *image* summary vectors Eq. (6)-(7) of [8] using **LIME** and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

Other details are the same as Section 3.6.

7 Comparison: LRP, Grad-GAM, and LIME

Compare three methods on text classification and image classification, addressing the aforementioned evaluation methods.

1. **Keep-K-only**: Select top K features (words, *visual* words or n-grams) using **LRP**, **Grad-GAM**, and **LIME** and check the classification accuracy under NBOW models after only keeping the selected features.

2. **Leave-K-out**: Check the classification accuracy under NBOW models after removing those selected features when feeding an input.

3. **KNN-based extrinsic validation method**: Compute two variants of *document* or *image* summary vectors Eq. (6)-(7) of [8] using **LRP**, **Grad-GAM**, and **LIME** and perform per-instance KNN-based *extrinsic validation* of Section 4.2 of [8].

7.1 Comparison results for text classification on large movie review dataset

Summarize the comparison results among LRP, Grad-GAM, and LIME for text classification under the settings of Keep-K-only, Leave-K-out, and KNN-based extrinsic validation method.

7.2 Comparison results for image classification on Pascal VOC 2009 dataset

Summarize the comparison results among LRP, Grad-GAM, and LIME for image classification under the settings of Keep-K-only, Leave-K-out, and KNN-based extrinsic validation method.

7.3 Discussion

Present the major observations on the comparison results and discuss some resulting issues.

8 Required submissions

For this project, you should submit the followings:

1. **Full reports** of addressing all the problems (derivation and summary)
2. **Complete source codes** (the main parts should mostly be written by yourself) of all the classification models [1] and explanation methods (LRP, Grad-GAM, and LIME) with proper comments.
3. **Complete additional codes and script**s (i.e, required for carrying out all the experiments).
4. **README files** for performing experiments (i.e., describe how to train and test your models). Prepare scripts and README files such that any user can easily execute training and testing on the dataset in most environments within 10 mins after reading the README file

References

[1] Yoshua Bengio. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2(1):1–127, January 2009.

[2] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[3] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. 521:436–444, 2015.

[4] Ning Xie, Gabrielle Ras, Marcel van Gerven, and Derek Doran. Explainable deep learning: A field guide for the uninitiated, 2020.

[5] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7), 2015.

[1] They should include NBOW and 1d convnet for text classification and the BoW model and the convolutional network for image classification

[6] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[7] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144, 2016.

[8] Leila Arras, Franziska Horn, Grégoire Montavon, Klaus-Robert Müller, and Wojciech Samek. "What is Relevant in a Text Document?": An Interpretable Machine Learning Approach. *PLOS ONE*, 12(8):e0181142, 2017.

[9] Moritz Böhle, Fabian Eitel, Martin Weygandt, and Kerstin Ritter. Layer-wise relevance propagation for explaining deep neural network decisions in mri-based alzheimer's disease classification. *Frontiers in Aging Neuroscience*, 11:194, 2019.

[10] Imran Sheikh, Irina Illina, Dominique Fohr, and Georges Linarès. Learning word importance with the neural bag-of-words model. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 222–229, 2016.

[11] Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. Understanding convolutional neural networks for text classification. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–65, 2018.

[12] Yu-Gang Jiang, Jun Yang, Chong-Wah Ngo, and Alexander G. Hauptmann. Representations of keypoint-based semantic concept detection: A comprehensive study. *IEEE Trans. Multim.*, 12(1):42–53, 2010.

[13] Chih-Fong Tsai. Bag-of-words representation in image annotation: A review. *International Scholarly Research Notices*, 2012, 2012.

[14] Piyawat Lertvittayakumjorn and Francesca Toni. Human-grounded evaluations of explanation methods for text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5195–5205, 2019.