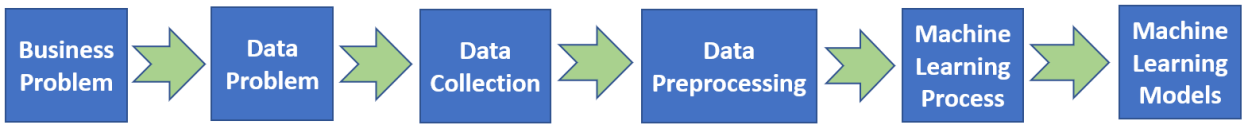
CSE353 – Machine Learning Course Projects Ideas

One of the highlights of CSE353 course is a course project which will involve concentrated work in one of the areas related to machine learning. Machine learning processes helps solving a particular business problem.



The possible topics of projects will be constrained to those listed below. It is expected that one group will take one project. You are free to investigate these topics and scope your problem to more specific problem. Each project has a "mentor" who is the first line of defense for individual discussions about your team's particular project. I will also be encouraging occasional "town halls" about each project during the class.

The project has following milestones:

1) **Milestone 1 (3 Sept 2020 – 2 Oct 2020): Problem definition, project objectives and literature search**:
   You will turn in 4-5 page of project proposal that consists of the following points:
   a. Problem definition: Which problem you are solving through this project? Why is it important to solve this problem?
   b. Project objectives: In order to solve above problem, how do you align the project objectives? Each project can have 3 – 4 clearly defined objectives.
   c. Literature search: Conduct a search of literature available in the scientific community to solve above problem or similar problems.
   d. Approach to solve the problem: Clearly mention what is your preferred approach/approaches to solve the problem.
   e. Each group will be responsible to turn in 4 - 5 pages project progress report for Milestone 1 as of the dates above. Milestone 1 has about 30% weightage. This is to encourage starting early, and to make sure that you and I both know what you are to do before it is too late to avoid trouble.

2) **Milestone 2 (3 Oct 2020 – 2 Nov 2020): Data collection and data preprocessing**:
   You will collect, clean and pre-process data necessary for your project. Your report will consist of the following points:
   a. Data requirements: Which data is required to solve the problem in hand?
   b. Data sources: Which data sources you have chosen and why?
   c. Data collection scripts: Did you write/use any scripts for data collection?
   d. Nature of raw data: What is the nature of raw data you have collected?

e. Data cleaning and preprocessing: Which data cleaning and preprocessing techniques you used on your data? Why?
f. Data repository: Create a private Github repository to store data necessary for your project and store raw and preprocessed data in this repository. Add your instructor and mentors as a collaborator.
g. Data visualization: Use Tableau or Python scripts to visualize the data you have preprocessed and submit a report.
h. Each group will be responsible to turn in 4 - 5 pages project progress report for Milestone 2 as of the dates above. Milestone 2 has about 30% weightage.

3) **Milestone 3 (3 Nov 2020 – 2 Dec 2020): Applying machine learning techniques and presenting your results**:
This is the final milestone of the project which will consist of applying machine learning techniques, testing and presenting your results for solving the problem in hand.
a. Machine learning techniques: Which machine learning techniques you have used to solve the problem at hand? Why these techniques were chosen? Choose around 3-4 different techniques so that you can compare your results with respect to applied algorithms. Also, if there is any other work which has solved the similar problem, compare your results with their as well.
b. Applying machine learning techniques: How did you apply the chosen machine learning techniques? Did you write any Python or similar scripts or used any of the available tools?
c. Presenting your results: Present your results in a tabular and graphical form. You can write scripts to create graphs or use any of the available tool.
d. Conclusion: What is the conclusion of your project? Have you been successful in achieving your project objectives? Provide a reflection on how your objectives have been fully met/partially met or unmet towards the end of the project.
e. Future work: Since you have now better understanding of the problem in hand, you can provide some directions for the future work, specially applying your results in a particular domain or exploring approaches to improve obtained results further.
f. Each group will be responsible to turn in about 10 pages of final project report as of the dates above. All the data and source code will be stored in a Github repository. The details will be shared with fellow instructors and mentors. Milestone 3 has about 40% weightage.

Final project presentations: Your final presentations will be held on Monday 07 Dec 12:30 PM – 03:00 PM (the end-term exam timeslot). We will use Zoom for presentations.

While I anticipate that much of the work will be done as the deadline approaches, it is important to get started early enough to discover insurmountable roadblocks in data acquisition or problem

definition before it is too late. The project proposal and progress reports have been instituted to ensure people get serious well in advance of the final deadline.

I hope that we can publish some of the best of your submissions as a conference/journal article. Choose a project which is challenging, creative, not obvious and possibly unexplored or unpublished by anyone else. Projects which solve some social problems, burning issues are welcome. In the due time, I will provide some more information about the projects such as grading criteria.

1. Classification and clustering of Covid-19 related information being made available on the SNS sites (Mentor: Pravin Pawar)

Millions of people worldwide contribute to already flooding information about Covid-19 on social networking sites such as Twitter, Facebook and Youtube. These posts could be classified into categories such as informative, hotspot alerts, hoax, medical treatment, economic assessment etc. Based on the analysis of existing posts, it should be possible to classify new posts which are being added every minute. You will focus on how to automatically classify Covid-19 related posts on one of the SNS sites such as Twitter, Facebook, news sites and make this classification available as a real-time service, such as web service. For the clustering problem, you will focus on clustering news articles or wiki articles related to Covid-19. It is similar to classification project above, except that the categories clusters are not known before-hand. By analyzing clusters, you can name the clusters into categories such as informative, hotspot alerts, hoax, medical treatment, economic assessment etc. Of course, it is not known which kind of clusters will emerge through such analysis.

- Refer to the following papers/articles to get you started:
- Cinelli, M., Quattrociocchi, W., Galeazzi, A., Valensise, C. M., Brugnoli, E., Schmidt, A. L., ... & Scala, A. (2020). The covid-19 social media infodemic. *arXiv preprint arXiv:2003.05004*.
- Thelwall, M., & Thelwall, S. (2020). Retweeting for COVID-19: Consensus building, information sharing, dissent, and lockdown life. *arXiv preprint arXiv:2004.02793*.
- Lucas, B., Elliot, B., & Landman, T. (2020). Online Information Search During COVID-19. *arXiv preprint arXiv:2004.07183*.
- Ferrara, E. (2020). # COVID-19 on Twitter: Bots, Conspiracies, and Social Media Activism. *arXiv preprint arXiv:2004.09531*.
- Harnessing Social Media for the COVID-19 Pandemic https://blogs.scientificamerican.com/observations/harnessing-social-media-for-the-covid-19-pandemic/
- Jahanbin, K., & Rahmanian, V. (2020). Using twitter and web news mining to predict COVID-19 outbreak. *Asian Pacific Journal of Tropical Medicine*, 13.
- Bullock, J., Pham, K. H., Lam, C. S. N., & Luengo-Oroz, M. (2020). Mapping the landscape of artificial intelligence applications against COVID-19. *arXiv preprint arXiv:2003.11336*.

2. Abstractive Text Summarization using Sequence to Sequence RNN and Sentiment Analysis of Covid-19 News Articles (Mentor: Pravin Pawar)

Abstractive summarization is the technique of generating a summary of a text from its main ideas, not by copying verbatim most salient sentences from text. Sentiment analysis is the interpretation and

classification of emotions (positive, negative and neutral) within text data using text analysis techniques. In this project, you will obtain Covid-19 related news articles from the Internet and perform abstractive text summarization using Attentional Encoder-Decoder Recurrent Neural Networks. Extractive summarization technuqies are also to be used for comparision. The summarized text articles will later be used for sentiment analysis.

Refer to the following papers/articles to get you started:

- Nallapati, R., Zhou, B., Gulcehre, C., & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.
- Shi, T., Keneshloo, Y., Ramakrishnan, N., & Reddy, C. K. (2018). Neural abstractive text summarization with sequence-to-sequence models. *arXiv preprint arXiv:1812.02303*.
- Hamzah, F., Lau C .H., Nazri, H., Ligot, D. V., Lee, G., Tan, C. L., . . . Salunga, R, E.(2020). CoronaTracker: World- wide COVID-19 Outbreak Data Analysis and Prediction. Bull World Health Organ, doi: http://dx.doi.org/10.2471/BLT.20.255695
- Bullock, J., Pham, K. H., Lam, C. S. N., & Luengo-Oroz, M. (2020). Mapping the landscape of artificial intelligence applications against COVID-19. *arXiv preprint arXiv:2003.11336*.

3. NLC2CMD (NeurIPS 2020 Competition) (Mentor: Byungkon Kang)

This project is one of the main tasks hosted by this year's NeurIPS competition track. In this task, you get to 'translate' natural language commands into Unix commands (actually, it's Bash syntax). The input is a natural language description of what the user wants to achieve in the Bash terminal, and the output is a corresponding Unix command. See Figure 1 for a concrete example. The training data consists of natural language queries extracted from StackOverflow, and can be downloaded from the project website: http://ibm.biz/nlc2cmd. A detailed description and training data are available through \compete" menu in the site. This competition will accept submissions until mid-October, so you probably won't be able to enter it, but at least you can compare your results with the winners!

```
clai$ how do i compress a directory into a bz2 file
Try >> tar -cjf <archive-file> <directory>
clai$ tar -cjf file.tar.bz2 nlc2cmd-webapp/
clai$ ls
Box     Documents       Pictures        file.tar.bz     nlc2cmd-webapp
CLAI    Downloads       Public          Desktop         Library
clai$ extract files from archive into a directory
Try >> tar -xf <archive-file> -C <directory>
clai$ tar -xf file.tar.bz2 -C temp/
tar: could not chdir to temp/
clai$ mkdir temp
clai$ tar -xf file.tar.bz2 -C temp/
clai$ grep for all files in a directory with "port" in it, show details
Try >> grep -rv "port" <directory>
clai$ grep for all files in a directory with "port" in it, show line numbers and match case
Try >> grep -rni "port" <directory>
clai$ grep -rni "port" ./temp/nlc2cmd-webapp/
./temp/nlc2cmd-webapp//run.py:4:imports
./temp/nlc2cmd-webapp//run.py:6:from flask import Flask, request, render_template
./temp/nlc2cmd-webapp//run.py:7:import json, os
./temp/nlc2cmd-webapp//run.py:9:from ibm_watson import AssistantV2
./temp/nlc2cmd-webapp//run.py:10:from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
./temp/nlc2cmd-webapp//run.py:85:    app.run(host='0.0.0.0', port=int(os.getenv('PORT', 3456)))
```

Figure 1: Screenshot from the competition website.

4. Fashion Compatibility (Mentor: Byungkon Kang)

This project involves the Polyvore data set [1], which is a collection of fashion out ts gathered into several 'valid' combinations. Along with the combination, the user has access to the corresponding images of each fashion item. The main task of the Polyvore project is to make the following two predictions.

- Fill in the blank (FITB): Given a set of combination and a blank, nd the best candidate for the blank that will be most compatible with the rest of the items.
- Outfit generation: Given a text or an image describing the current situation, recommend a fashion item (or preferably a combination of them) that best suits the occasion.
- Fashion compatibility prediction: Given a set of fashion items, output a value in [0; 1] that quantitatively describes how fitting the combination is.

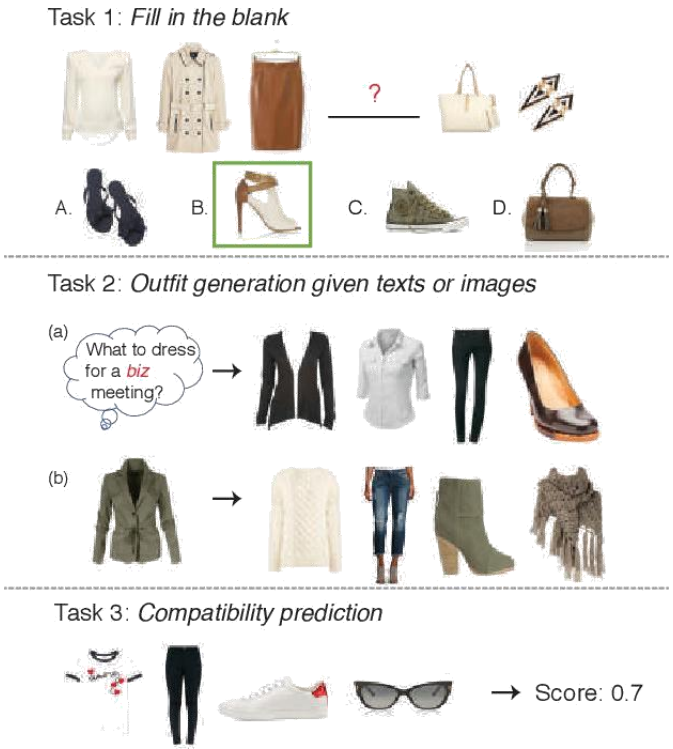


Figure 2: Task description. Image from [1].

See Figure 2 for a visual demonstration of these tasks. For a more detailed data set description, visit https://github.com/xthan/polyvore-dataset.

References

[1] Xintong Han, Zuxuan Wu, Yu-Gang Jiang, and Larry S Davis. Learning fashion compatibility with bidirectional lstms. In ACM Multimedia, 2017.

5. Text-Image Similarity (Mentor: Byungkon Kang)

This is a task of measuring the semantic similarity between a given image and a sentence. The similarity that we measure is a non-negative real number between 0 and 1. For example, if the image is a picture of Leonardo Da Vinci's Mona Lisa, and the accompanying sentence reads \A gentleman with a subtle smile", then the similarity score should be somewhat low (but not too low).

Formally, you should train a function $S_\delta : I \times L \to [0; 1]$, where I is the set of all possible images, and L is the set of all possible English sentences. The trainable parameter δ can be anything: the weights for the neural network, or the coefficients for the SVM, etc. The challenging part of this task is that not only should the model score high for relevant pairs, it should score low for irrelevant pairs as well.

For the training data, you can use whatever image-text pair you can get your hands on, but if you're out of time, I suggest using MSCOCO ([1], https://cocodataset.org/#home) and/or Flickr30k ([2], https://github.com/ BryanPlummer/flickr30k_entities). These data sets provide five different captions per each image in the set, so you'll have some amount of diversity.

References

[1] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollr. Microsoft COCO: Common objects in context. In Proceedings of ECCV, 2014.

[2] Bryan A. Plummer, Liwei Wang, Christopher M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. IJCV, 123(1):74{93, 2017.

6. Classification of Chest X-ray/CT-Scan databases for identification of Covid-19 (Mentor: Deepak Pradhan, ARKRAY Inc.)

Most people who get COVID-19 have mild or moderate symptoms like coughing, a fever, and shortness of breath. But some who catch the new coronavirus get severe pneumonia in both lungs. COVID-19 pneumonia is a serious illness that can be deadly. The aim of this project is to separate COVID-19 pneumonia patients from normal patients or non-Covid-19 pneumonia patients by analyzing digital x-ray/CT-scan images. There exist many Chest X-Ray/CT-scan databases and approaches for solving this problem.

Refer to the following papers/articles to get you started:

Covid-19 Chest-X-ray database: https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

COVID-CT-Dataset: A CT Scan Dataset about COVID-19: https://arxiv.org/abs/2003.13865

- Chowdhury, M. E., Rahman, T., Khandakar, A., Mazhar, R., Kadir, M. A., Mahbub, Z. B., ... & Reaz, M. B. I. (2020). Can AI help in screening Viral and COVID-19 pneumonia?. *arXiv preprint arXiv:2003.13145*.
- Cohen, J. P., Morrison, P., & Dao, L. (2020). COVID-19 image data collection. *arXiv preprint arXiv:2003.11597*.

- Narin, A., Kaya, C., & Pamuk, Z. (2020). Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks. *arXiv preprint arXiv:2003.10849*.
- Amyar, A., Modzelewski, R., & Ruan, S. (2020). Multi-task Deep Learning Based CT Imaging Analysis For COVID-19: Classification and Segmentation. *medRxiv*.
- Yoon, S. H., Lee, K. H., Kim, J. Y., Lee, Y. K., Ko, H., Kim, K. H., ... & Kim, Y. H. (2020). Chest radiographic and CT findings of the 2019 novel coronavirus disease (COVID-19): analysis of nine patients treated in Korea. *Korean journal of radiology*, *21*(4), 494-500.
- Khan, A. I., Shah, J. L., & Bhat, M. (2020). CoroNet: A Deep Neural Network for Detection and Diagnosis of Covid-19 from Chest X-ray Images. *arXiv preprint arXiv:2004.04931*.
- Zhao, J., Zhang, Y., He, X., & Xie, P. (2020). COVID-CT-Dataset: a CT scan dataset about COVID-19. *arXiv preprint arXiv:2003.13865*.

7. 3D Machine Learning for Gesture and Pose Recognition (Mentor: Homin Park, Prof. Wesley De Neve, Ghent University Global Campus)

3-D machine learning is a technology that reproduces the 3-D space and objects from the real world, making it subsequently possible to extract all kinds of information from this reproduction. It is technology that is rapidly changing our lives. For example, a service has been recently announced that scans a human body to create a virtual avatar, for instance facilitating virtual clothes fitting. 3-D machine learning is also used as a core technology to recognize space in robot vacuum cleaners and to guide service robots at Incheon Airport. The technology is also applied to driverless/driving assistant systems.

Basically, for 3-D machine learning, we need to acquire 3-D data by combining expensive sensors or by making use of computationally expensive 2-D images. The need for a lot of time and money to build an acquisition system has slowed down 3-D machine learning research. However, given that requirements for computer games have increased recently, cheap and good home sensors have begun to appear in the market, such as Microsoft Kinect, Intel RealSense, and so on.

In this project, the basic idea is to make use of 3-D cameras to acquire a 3-D point cloud of the human body structure (skeleton), and to subsequently create one or more machine learning models that are able to classify gestures and/or pose, following by analysis and synthesis of the experimental results obtained. Next to the basic idea presented, other ideas making use of 3-D cameras are welcome.

Industry mentored topics (Mentor: Researchers at Solidware Da Vinci Labs Korea https://davincilabs.ai/en)

Multiple topics are available which will be mentored by researchers from Solidware, Korea. Depending on your interest, we can get in touch with Solidware.

8. Anomaly detection in time-series (practical oriented)

Anomaly detection problem for time series is usually formulated as finding outlier data points relative to some standard or usual signal. For example, when your server goes down and you see zero or a really low number of users for some short period of time. These types of anomalies are usually classified as temporal changes. Basically, an anomaly detection algorithm should either label each time point with anomaly/not anomaly, or forecast a signal for some point and test if this point value varies from the forecasted enough to deem it as an anomaly. Datasets available at: http://odds.cs.stonybrook.edu/#table3

Dataset	Type	Size	Duration	Description
DataMarket - TSDL	Univariate	Multiple datasets	--	The Time Series Data Library (TSDL) was created by Rob Hyndman, Professor of Statistics at Monash University, Australia.
Yahoo - a benchmark dataset for TSAD	Multivariate	between 741 and 1680 observations per series at regular interval	367 time series	This dataset is released by Yahoo Labs to detect unusual traffic on Yahoo servers.
Numenta Anomaly Benchmark (NAB)	Multivariate	Multiple datasets	--	Numenta Anomaly Benchmark, a benchmark for streaming anomaly detection where sensor provided time-series data is utilized.

9. MNIST classification using active learning (practical oriented)

Active learning is a special case of machine learning in which a learning algorithm can interactively query a user (or some other information source) to label new data points with the desired outputs. There are situations in which unlabeled data is abundant but manual labeling is expensive. In such a scenario, learning algorithms can actively query the user/teacher for labels. This type of iterative supervised learning is called active learning. Since the learner chooses the examples, the number of examples to learn a concept can often be much lower than the number required in normal supervised learning. You will use active learning for The MNIST database of handwritten digits, which has a training set of 60,000 examples, and a test set of 10,000 examples (http://yann.lecun.com/exdb/mnist/). The digits have been size-normalized and centered in a fixed-size image.

10. Classification using GAN for augmentation of tabular data - usecase of credit card fraud detection (https://www.kaggle.com/mlg-ulb/creditcardfraud - research oriented)

It is a difficult task to classify records with multiple labels only using a small number of labeled samples and to be worse, with unbalanced distribution. Generative adversarial networks (GANs) can be used for data augmentation which is able to complement and complete the data manifold from the true sense, assist the classifier to better find margins or hyper-planes of neighboring classes, and finally lead to better performance in a classification task. You will apply this technique for credit card fraud detection dataset that contains transactions made by credit cards in September 2013 by European cardholders. This dataset

presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

11. An ML-powered autocomplete feature using RoBERTa model (Research Oriented https://github.com/pytorch/fairseq/tree/master/examples/roberta)

Language model pretraining has led to significant performance gains but careful comparison between different approaches is challenging. Training is computationally expensive, often done on private datasets of different sizes, and, as we will show, hyperparameter choices have significant impact on the final results. We present a replication study of BERT pretraining (Devlin et al., 2019) that carefully measures the impact of many key hyperparameters and training data size. We find that BERT was significantly undertrained, and can match or exceed the performance of every model published after it. Our best model achieves state-of-the-art results on GLUE, RACE and SQuAD. These results highlight the importance of previously overlooked design choices, and raise questions about the source of recently reported improvements. We release our models and code.

12. A predictive text generator using open ai's GPT-2 (Research Oriented https://github.com/openai/gpt-2)

GPT-2 is a large-scale unsupervised language model which generates coherent paragraphs of text, achieves state-of-the-art performance on many language modeling benchmarks, and performs rudimentary reading comprehension, machine translation, question answering, and summarization—all without task-specific training. GPT-2 generates synthetic text samples in response to the model being primed with an arbitrary input. The model is chameleon-like—it adapts to the style and content of the conditioning text. This allows the user to generate realistic and coherent continuations about a topic of their choosing.

13. Single-layer approximation of multi-layer bidirectional RNNs (Research Oriented https://arxiv.org/pdf/1909.00021.pdf)

Recent work has shown that topological enhancements to recurrent neural networks (RNNs) can increase their expressiveness and representational capacity. Two popular enhancements are stacked RNNs, which increases the capacity for learning non-linear functions, and bidirectional processing, which exploits acausal information in a sequence. In this work, we explore the delayed-RNN, which is a single-layer RNN that has a delay between the input and output. We prove that a weight-constrained version of the delayed-RNN is equivalent to a stacked-RNN. We also show that the delay gives rise to partial acausality, much like bidirectional networks. Synthetic experiments confirm that the delayedRNN can mimic bidirectional networks, solving some acausal tasks similarly, and outperforming them in others. Moreover, we show similar performance to bidirectional networks in a realworld natural language processing task. These results suggest that delayed-RNNs can approximate topologies including stacked RNNs, bidirectional RNNs, and stacked bidirectional RNNs – but with equivalent or faster runtimes for the delayed-RNNs.

14. Compact RNN - combination of CP and Tucker for speedup (also Tensor Regression Networks – Research Oriented)

http://openaccess.thecvf.com/content_cvpr_2018/papers/Ye_Learning_Compact_Recurrent_CVPR_2018_paper.pdf

Recurrent Neural Networks (RNNs) are powerful sequence modeling tools. However, when dealing with high dimensional inputs, the training of RNNs becomes computational expensive due to the large number of model parameters. This hinders RNNs from solving many important computer vision tasks, such as Action Recognition in Videos and Image Captioning. To overcome this problem, we propose a compact and flexible structure, namely Block-Term tensor decomposition, which greatly reduces the parameters of RNNs and improves their training efficiency. Compared with alternative low-rank approximations, such as tensortrain RNN (TT-RNN), our method, Block-Term RNN (BTRNN), is not only more concise (when using the same rank), but also able to attain a better approximation to the original RNNs with much fewer parameters. On three challenging tasks, including Action Recognition in Videos, Image Captioning and Image Generation, BT-RNN outperforms TT-RNN and the standard RNN in terms of both prediction accuracy and convergence rate. Specifically, BT-LSTM utilizes 17,388 times fewer parameters than the standard LSTM to achieve an accuracy improvement over 15.6% in the Action Recognition task on the UCF11 dataset.

15. Recurrent highway networks (https://arxiv.org/abs/1607.03474 - Research Oriented)

Many sequential processing tasks require complex nonlinear transition functions from one step to the next. However, recurrent neural networks with 'deep' transition functions remain difficult to train, even when using Long Short-Term Memory (LSTM) networks. We introduce a novel theoretical analysis of recurrent networks based on Gersgorin's circle theorem that illuminates several modeling and optimization issues and improves our understanding of the LSTM cell. Based on this analysis we propose Recurrent Highway Networks, which extend the LSTM architecture to allow step-to-step transition depths larger than one. Several language modeling experiments demonstrate that the proposed architecture results in powerful and efficient models. On the Penn Treebank corpus, solely increasing the transition depth from 1 to 10 improves word-level perplexity from 90.6 to 65.4 using the same number of parameters. On the larger Wikipedia datasets for character prediction (text8 and enwik8), RHNs outperform all previous results and achieve an entropy of 1.27 bits per character.

16. General network compression (to assess advantages – Research Oriented)

Neural networks are both computationally intensive and memory intensive, making them difficult to deploy on embedded systems with limited hardware resources. To address this limitation, we introduce “deep compression”, a three stage pipeline: pruning, trained quantization and Huffman coding, that work together to reduce the storage requirement of neural networks by 35× to 49× without affecting their accuracy. Our method first prunes the network by learning only the important connections. Next, we quantize the weights to enforce weight sharing, finally, we apply Huffman coding. After the first two steps we retrain the network to fine tune the remaining connections and the quantized centroids. Pruning, reduces the number of connections by 9× to 13×; Quantization then reduces the number of bits that represent each connection from 32 to 5. On the ImageNet dataset, our method reduced the storage required by AlexNet by 35×, from 240MB to 6.9MB, without loss of accuracy. Our method reduced the size of VGG-16 by 49× from 552MB to 11.3MB, again with no loss of accuracy. This allows fitting the model

into on-chip SRAM cache rather than off-chip DRAM memory. Our compression method also facilitates the use of complex neural networks in mobile applications where application size and download bandwidth are constrained. Benchmarked on CPU, GPU and mobile GPU, compressed network has 3× to 4× layerwise speedup and 3× to 7× better energy efficiency.

References

[1] landmark paper (https://arxiv.org/pdf/1510.00149.pdf)
[2] what is the state of neural network pruning? (paper: https://arxiv.org/pdf/2003.03033.pdf)
[3] Tensorflow examples

a. https://www.tensorflow.org/model_optimization/guide/quantization/post_training

b. https://www.tensorflow.org/model_optimization/guide/quantization/training_comprehensive_guide

17. Self-attention (Research Oriented)

Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. It has been shown to be very useful in machine reading, abstractive summarization, or image description generation. The long short-term memory network paper used self-attention to do machine reading. In the example below, the self-attention mechanism enables us to learn the correlation between the current words and the previous part of the sentence.

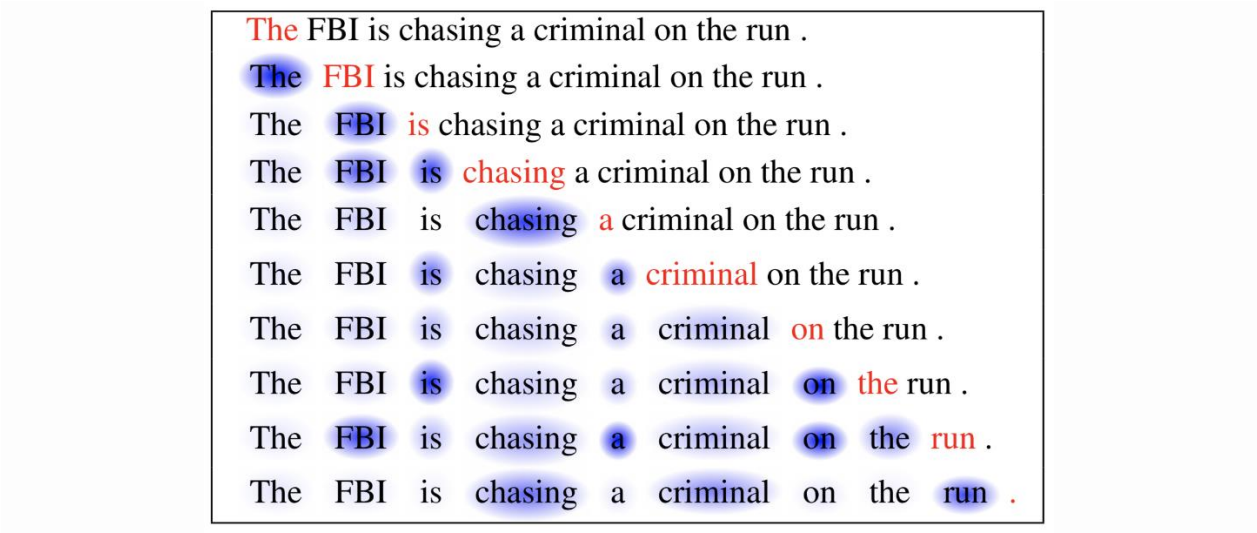


Fig. 3. The current word is in red and the size of the blue shade indicates the activation level. (Image source: Cheng et al., 2016*)*

Reference

(Cheng et. al. 2016): Cheng, Jianpeng, Li Dong, and Mirella Lapata. "Long short-term memory-networks for machine reading." *arXiv preprint arXiv:1601.06733* (2016).

18. Modeling using Transformers

The Transformer architecture is superior to RNN-based models in computational efficiency. Recently, GPT and BERT demonstrate the efficacy of Transformer models on various NLP tasks using pre-trained language models on large-scale corpora. Surprisingly, these Transformer architectures are suboptimal for language model itself. Neither self-attention nor the positional encoding in the Transformer is able to efficiently incorporate the word-level sequential context crucial to language modeling. In this paper, we explore effective Transformer architectures for language model, including adding additional LSTM layers to better capture the sequential context while still keeping the computation efficient. We propose Coordinate Architecture Search (CAS) to find an effective architecture through iterative refinement of the model. Experimental results on the PTB, WikiText-2, and WikiText-103 show that CAS achieves perplexities between 20.42 and 34.11 on all problems, i.e. on average an improvement of 12.0 perplexity units compared to state-of-the-art LSTMs. The source code is publicly available.

Reference

(Wang C. et. al., 2019): Wang, C., Li, M., & Smola, A. J. (2019). Language models with transformers. *arXiv preprint arXiv:1904.09408*.