Recurrent Neural Network vs. Cascaded Recurrent Neural Network for Land Cover Classification

Iryna Repinetska,
HU Berlin,
Institut für Informatik

Overview

1. Project Description
2. Houston Dataset
3. Recurrent Neural Network (RNN)
   a. Idea behind RNN
   b. RNN for Land Cover Classification
   c. Hyperparameters for RNN
   d. Experiments for RNN
4. Cascaded Recurrent Neural Network (CasRNN)
   a. Basic Structure
   b. Hyperparameters for CasRNN
   c. Experiments for CasRNN
5. Comparison of the models
6. Conclusion
7.

Project Description:

- Given:
  - Code for:
    - RNN using GRU;
    - CasRNN using GRU;
    - PyTorch.
  - Houston Dataset.
- To do:
  - Investigate the influence of hyperparameters;
  - Compare the processing time (training and inference);
  - Compare the accuracies:
    - Overall accuracy;
    - Average accuracy;
    - Cohen's Kappa;
  - Produce classification maps.

Houston Dataset

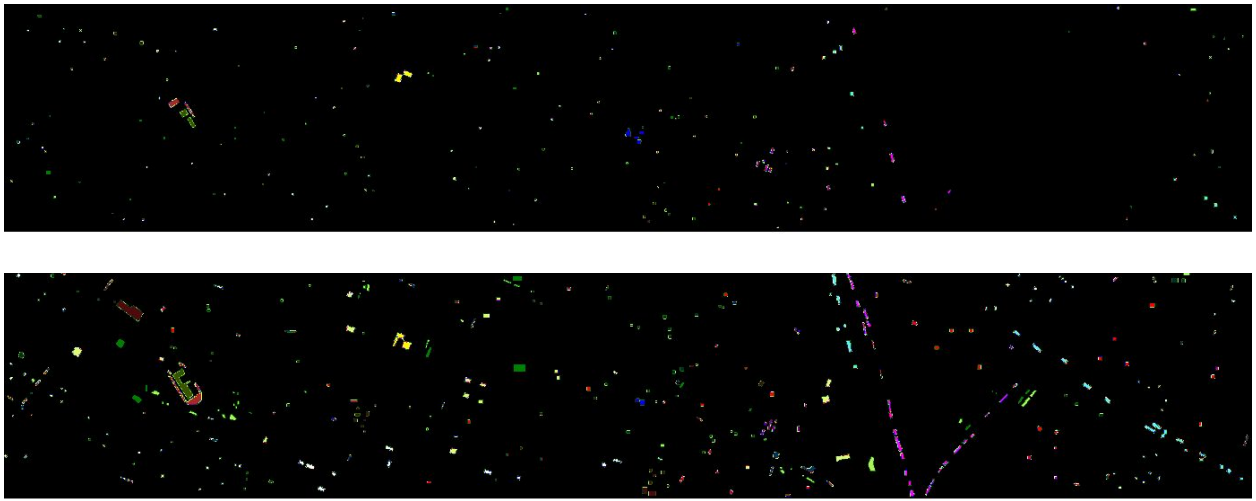
- 15 classes;
- 2832 training samples;
- 12179 test samples;
- 180 - 200 samples for a class;
- Special dimensions are 349 × 1905;
- 144 spectral bands;
- the area of University of Houston campus and the neighboring urban area;
- the three-band false color composite:

Class name	Training samples	Test samples	Total samples
Healthy grass	198	1053	1251
Stressed grass	190	1064	1254
Synthetic grass	192	505	697
Trees	188	1056	1244
Soil	186	1056	1242
Water	182	143	325
Residential	196	1072	1268
Commercial	191	1053	1244
Road	193	1059	1252
Highway	191	1036	1227
Railway	181	1054	1235
Parking Lot 1	192	1042	1234
Parking Lot 2	184	285	469
Tennis Court	181	247	428
Running Track	187	473	660
Total	2832	12179	15011

Li, Shutao & Song, Weiwei & Fang, Leyuan & Chen, Yushi & Ghamisi, Pedram & Benediktsson, Jon. (2019). Deep Learning for Hyperspectral Image Classification: An Overview. IEEE Transactions on Geoscience and Remote Sensing. PP. 1-20. 10.1109/TGRS.2019.2907932.

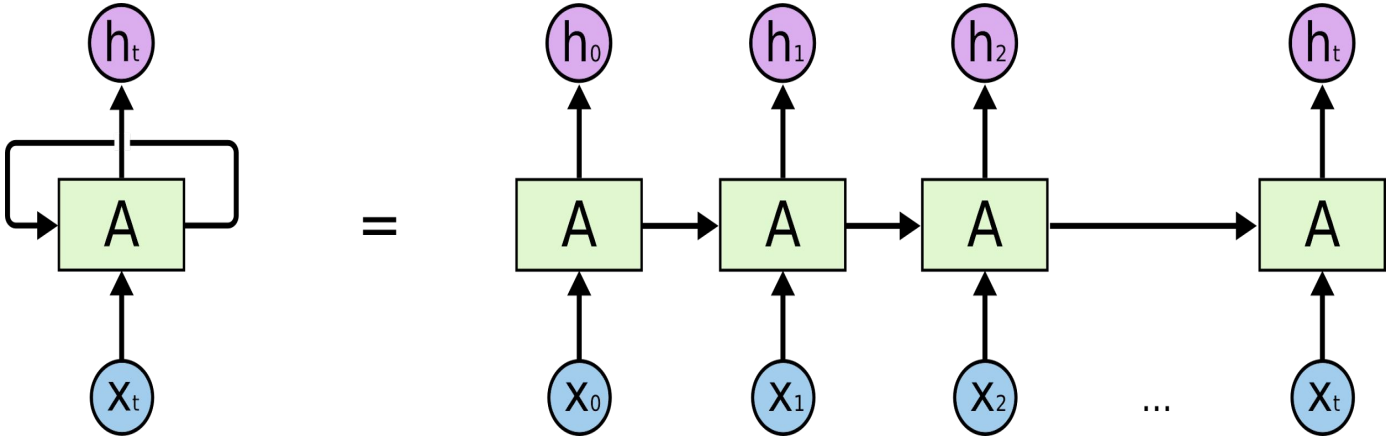
Training, Validation, Testing data

- The initial training data was split at random into training set (0.9%, 2548 samples) and validation set (0.1%, 284 samples).
- validation set was used for hyperparameter tuning;
- test data was used only for the final evaluation when the best hyperparameters were chosen;
- top:
  initial training data;
- bottom:
  test set.



Recurrent Neural Network (RNN)

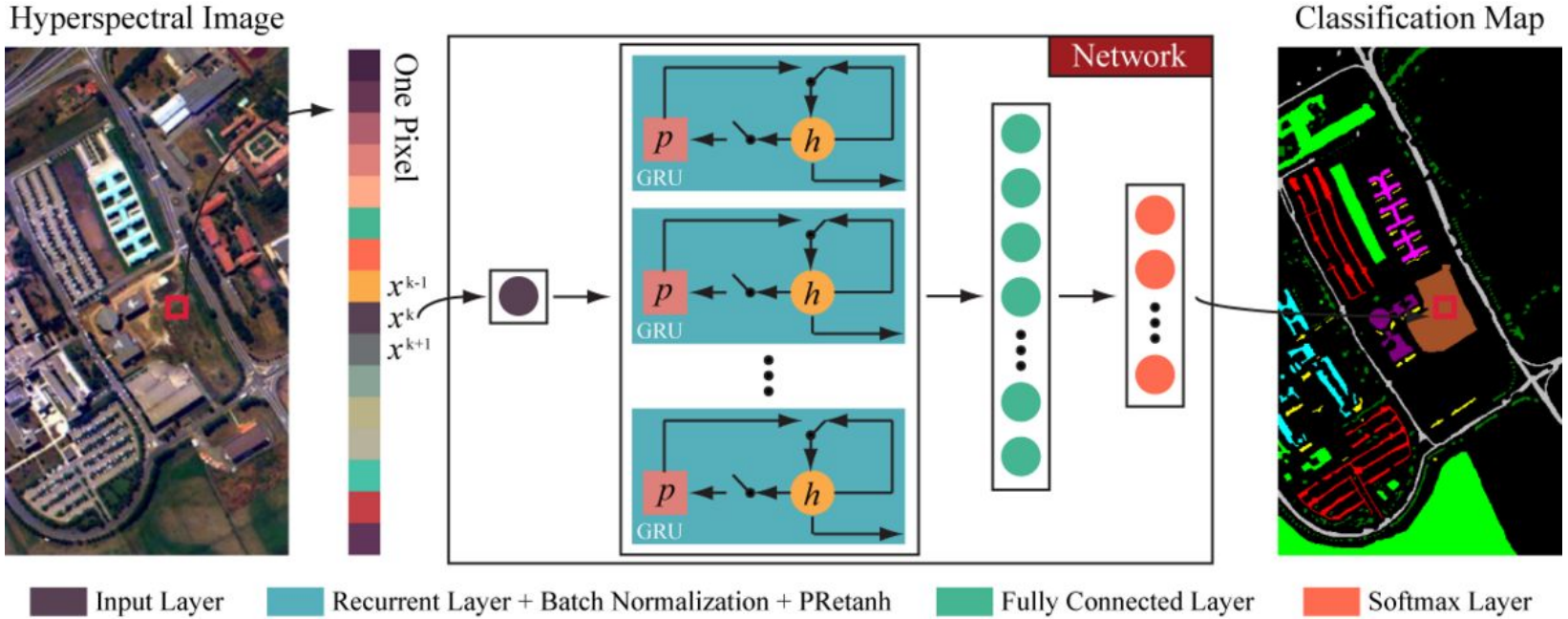
- sequential data;
- looks at the current input and gets information from the previous inputs (memory);
- Gated Recurrent Unit:
  - adresses vanishing gradient problem;
  - has less complex structure than Long Short Term Memory (LSTM) ⟹
  - computationally more efficient and more suitable for less data.



https://www.knime.com/blog/text-generation-with-lstm

RNN for Land Cover Classification

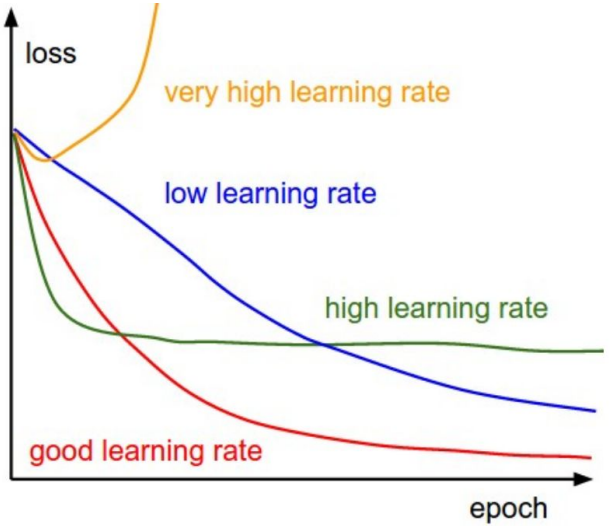
- analyze hyperspectral pixel as sequential data
- apply to all pixels of the image



L. Mou, P. Ghamisi and X. X. Zhu, "Deep Recurrent Neural Networks for Hyperspectral Image Classification," in *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 7, pp. 3639-3655, July 2017, doi: 10.1109/TGRS.2016.2636241.

Hyperparameters

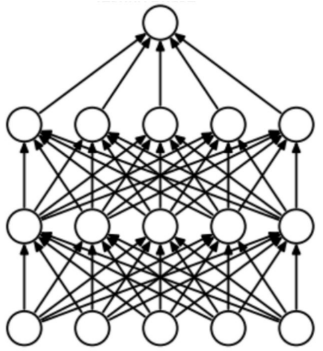
- not RNN specific:
  - Learning rate:
    - controls how much weights of the network are adjusted with respect to the loss gradient;
    - too low, then the training will take a lot of time and resources;
    - too high, then loss may not converge or even diverge.
  - Number of Epochs:
    - epoch is one forward pass and one backward pass of all the training examples;
    - depends on learning rate.



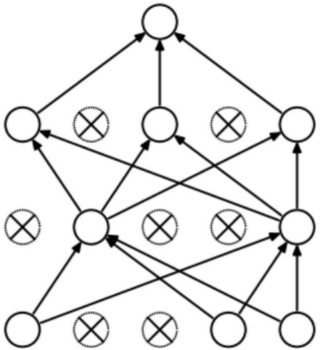
https://cs231n.github.io/neural-networks-3/

Hyperparameters

- <u>not RNN specific:</u>
  - Dropout:
    - *to prevent overfitting*: helps to reduce interdependent learning amongst the neurons, type of regularization in neural networks;
    - forces a neural network to learn more robust features.
    - *at each training stage*: individual nodes are either dropped out of the net with probability *1-p* or kept with probability *p*, a reduced network is left; incoming and outgoing edges to a dropped-out node are also removed;
    - *testing phase*: use all activations, but reduce them by a factor *p* (to account for the missing activations during training.



(a) Standard Neural Net

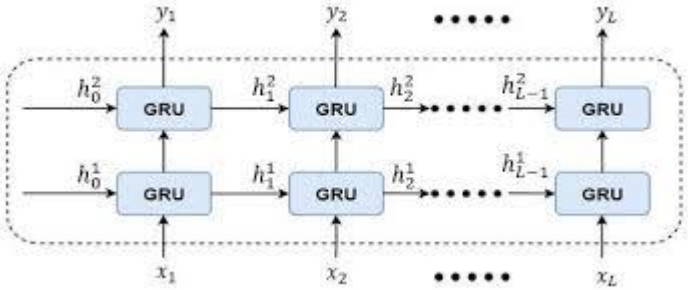


(b) After applying dropout.

https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5

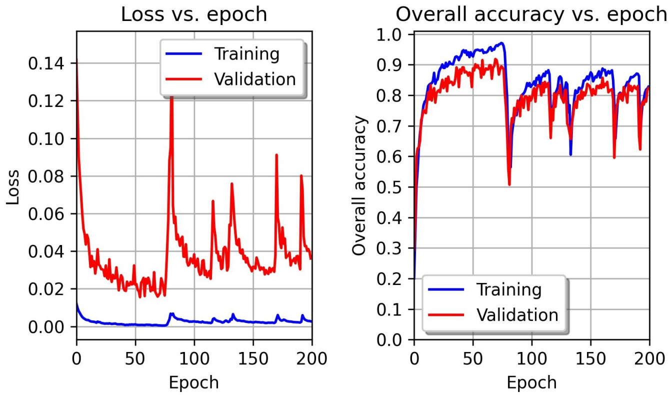
Hyperparameters

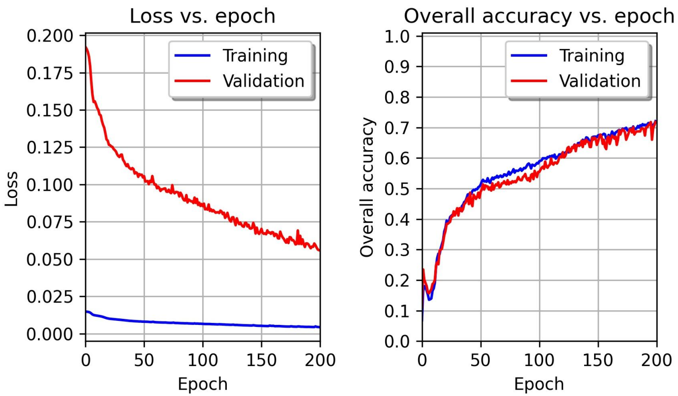
- RNN specific:
  - Size of RNN (hidden size)
    - the number of features in the hidden state;
  - Number of Layers for RNN
    - the number of recurrent layers;
  - Dropout
    - introduces a dropout layer on the outputs of each RNN layer except the last layer.

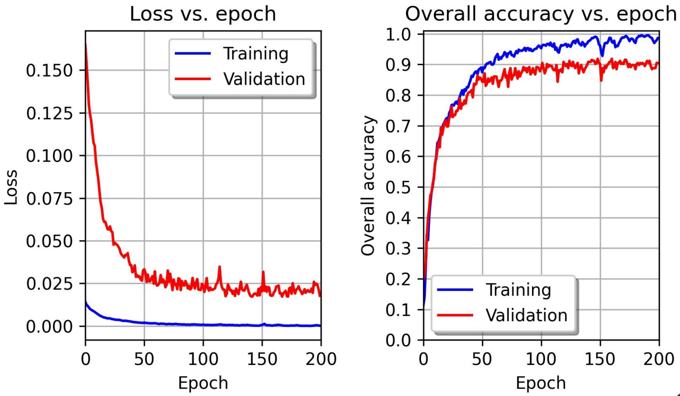


Experiments for RNN

- Learning rate:
  - lr = {0.0001, 0.001, 0.01}
    - 0.0001 bottom left;
    - 0.001 bottom right;
    - 0.01 top right;
  - hidden size = 128;
  - number of layers = 2;
  - dropout = 0.

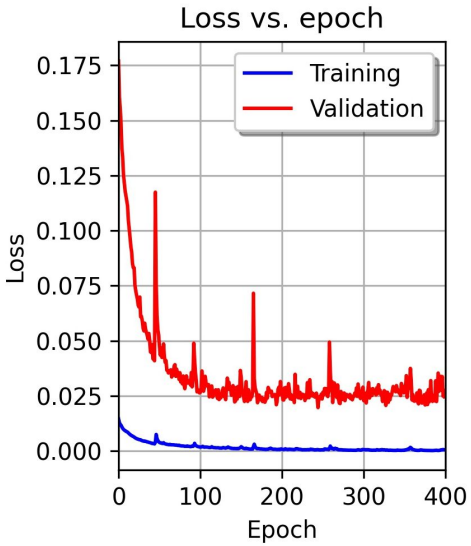


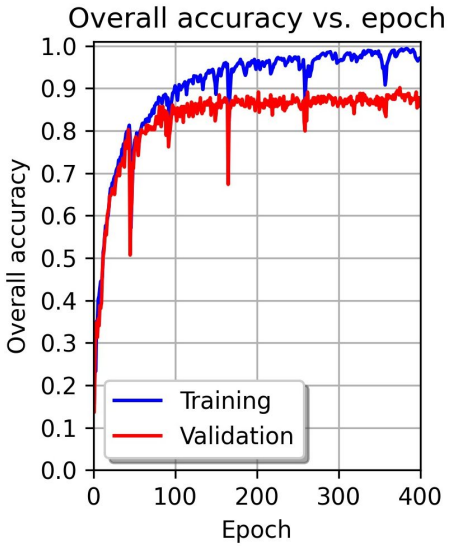




Experiments for RNN

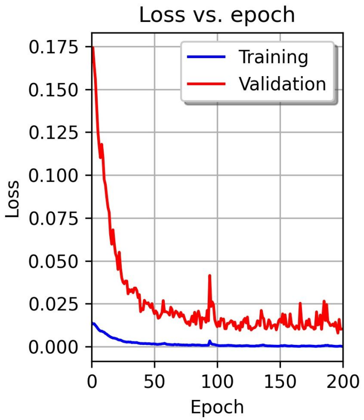
- Number of epochs:
  - lr = 0.001;
  - hidden size = 128;
  - number of layers = 2;
  - dropout = 0.
  - Observation:
    - validation loss does not decrease after 200 epoch ⟹
    - fix 200 epochs.

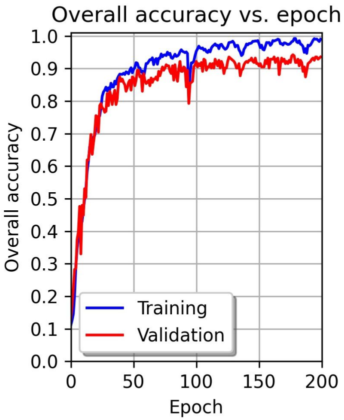


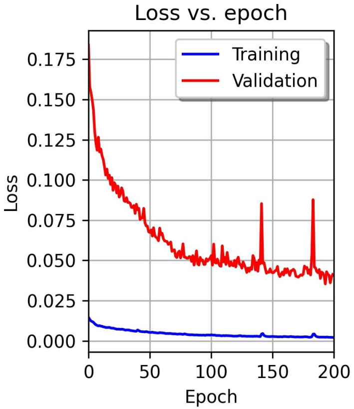


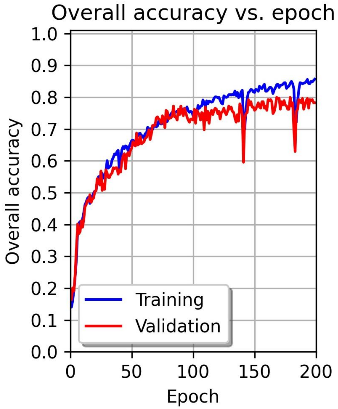
Experiments for RNN

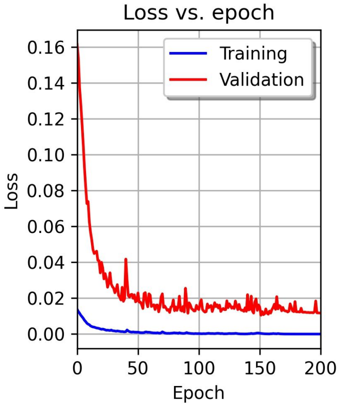
- Number of layers:
  - number layers = {1, 2, …, 12}
    - 1 bottom left; 2 bottom right; 12 top right;
  - lr = 0.001;
  - epoch = 200;
  - hidden size = 128;
  - dropout = 0.

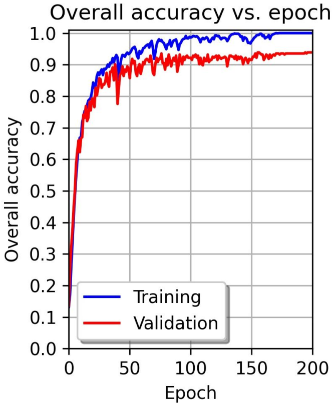






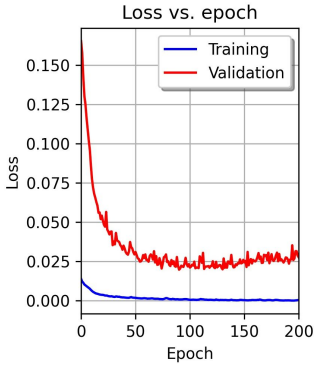


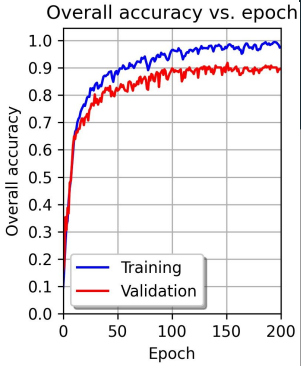




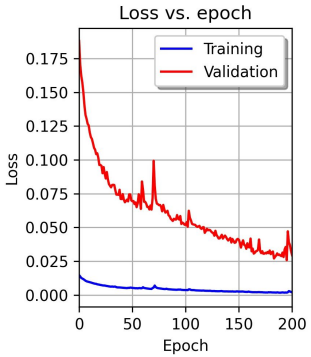
Experiments for RNN

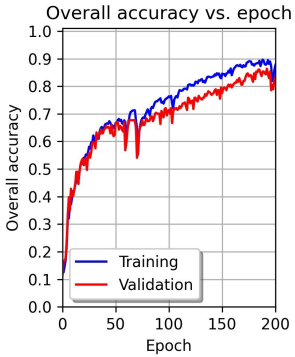
- Hidden size:
  - first iteration
    - hidden size = {50, 100, 150, 200, 250}
      - 50 bottom left;
      - 100 bottom right;
      - 150 top right.

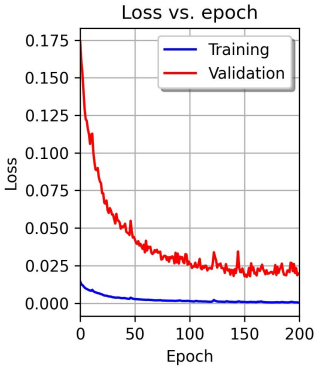


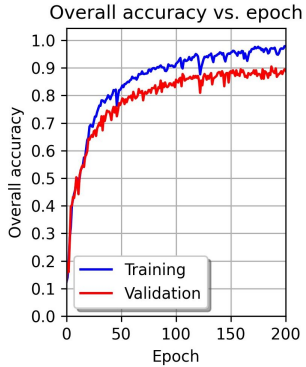


  - lr = 0.001;
  - epoch = 200;
  - num layers = 2;
  - dropout = 0.



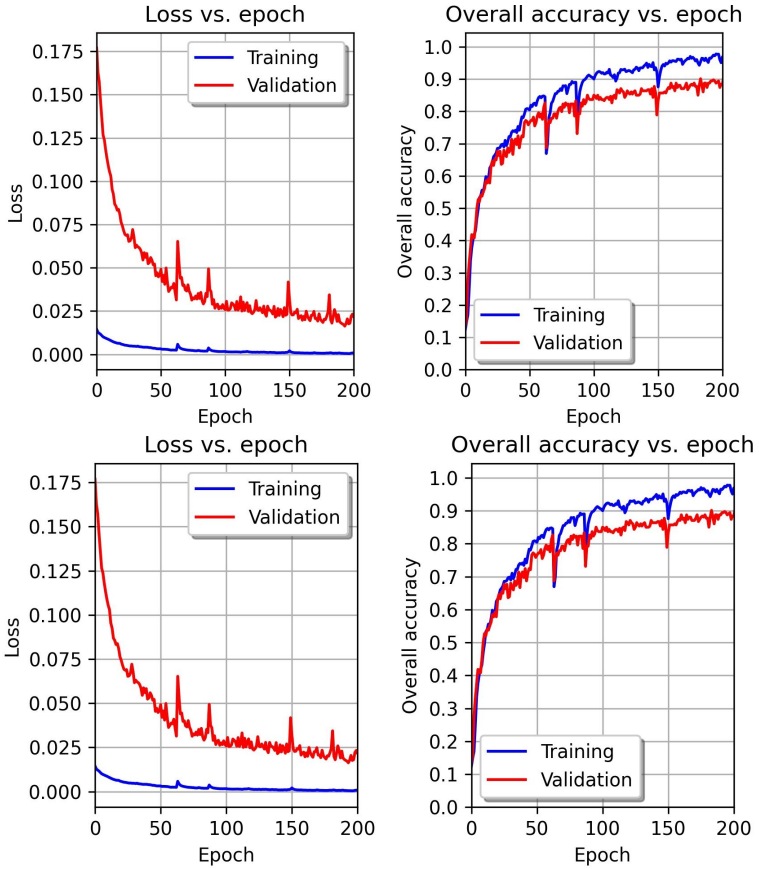






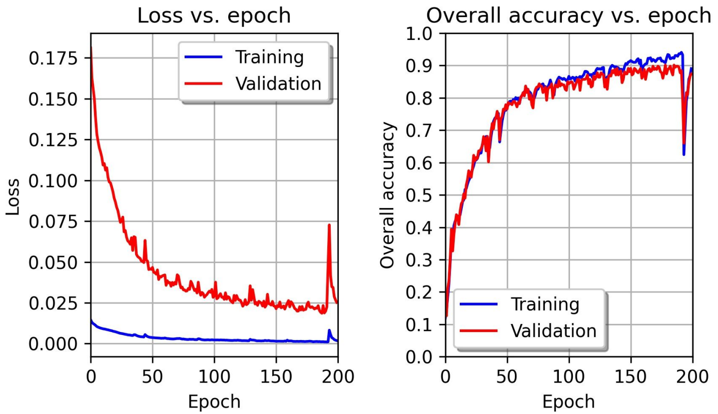
Experiments for RNN

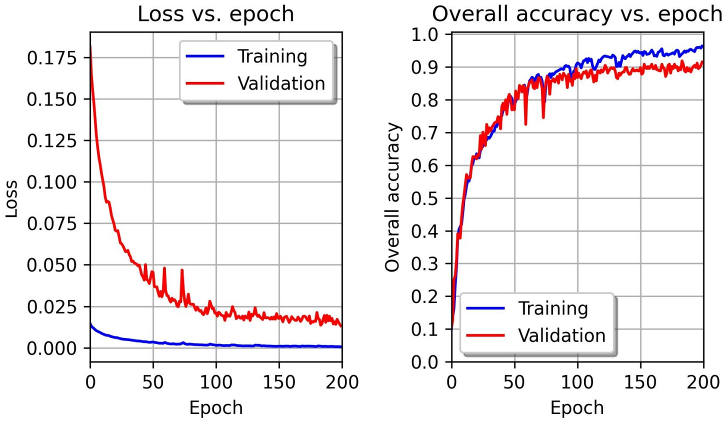
- Hidden size:
  - second iteration
    - hidden size = {80, 110, 120}
      - 80 bottom right;
      - 120 top right.
  - lr = 0.001;
  - epoch = 200;
  - number layers = 2;
  - dropout = 0.

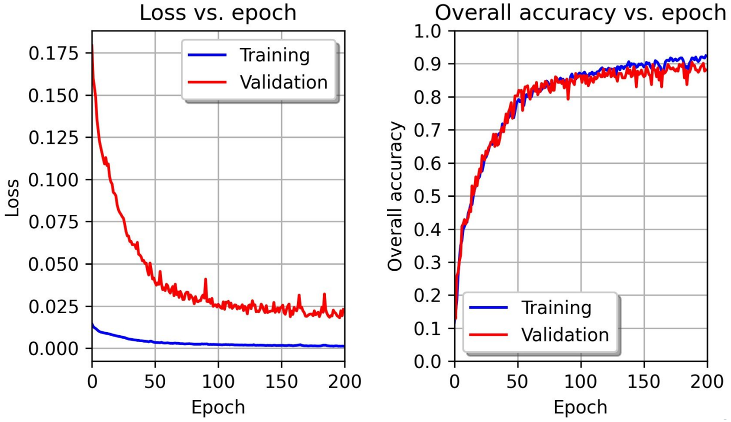


Experiments for RNN

- Dropout:
  - dropout = {0, 0.3, 0.5, 0.7}
    - 0.3 bottom left; 0.5 bottom right; 0.7 top;
  - lr = 0.001;
  - epoch = 200;
  - num layers = 2;
  - hidden size = 80.

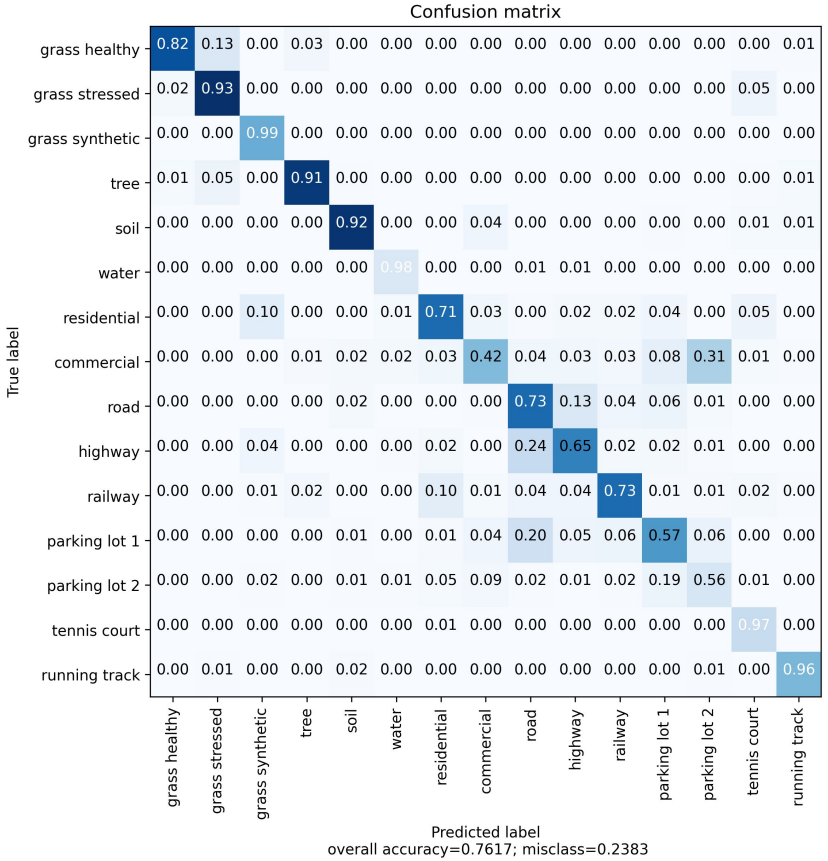




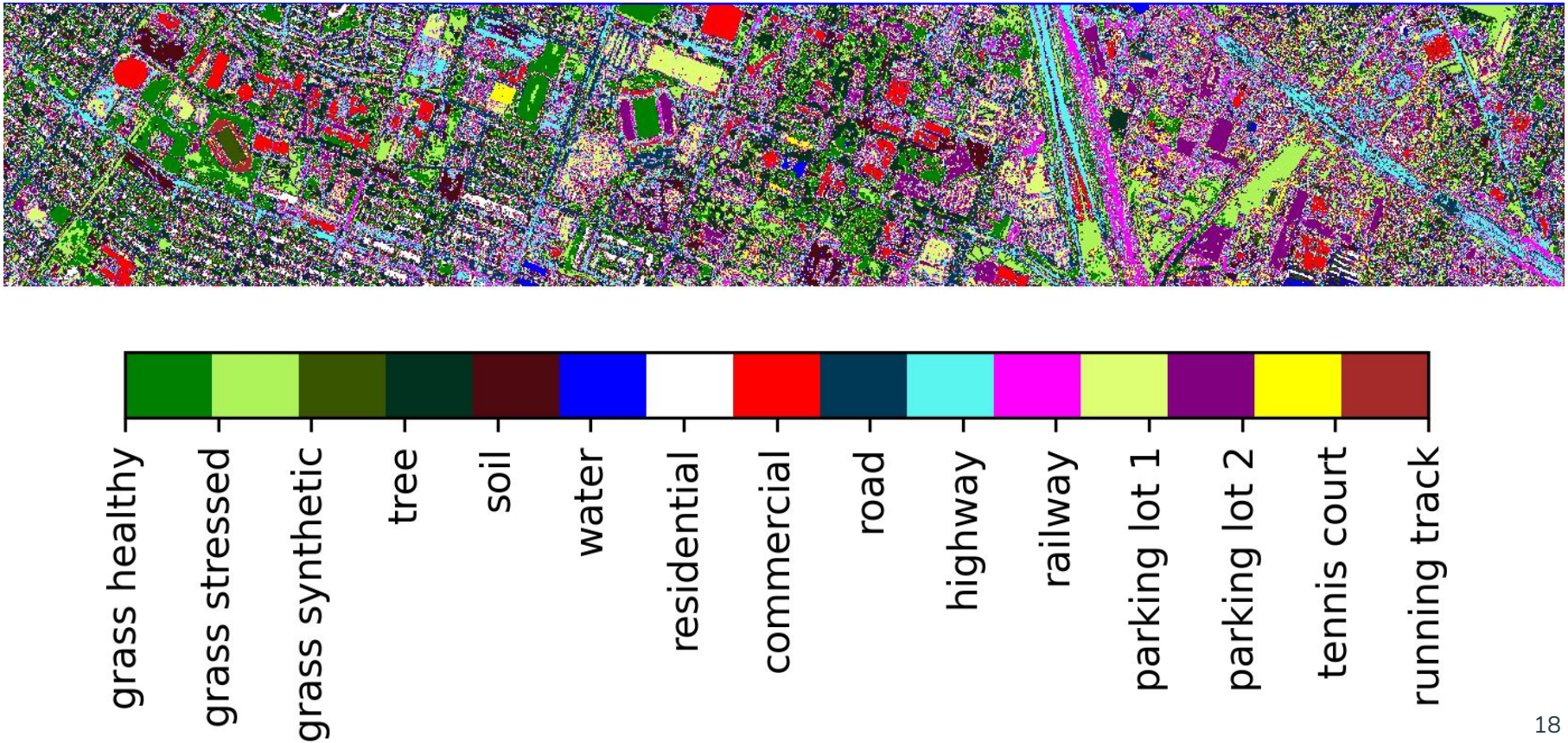


Confusion matrix for RNN

- Final parameters:
  - lr = 0.001;
  - epoch = 200;
  - num layers = 2;
  - hidden size = 80;
  - dropout = 0.3,
- Best prediction accuracy:
  - water - 0.98%;
  - grass stressed - 99%;
  - tennis court - 97%;
- Worst prediction accuracy:
  - commercial - 42%;
    - parking lot 2 - 31%.
  - highway - 65%
    - road - 24%.

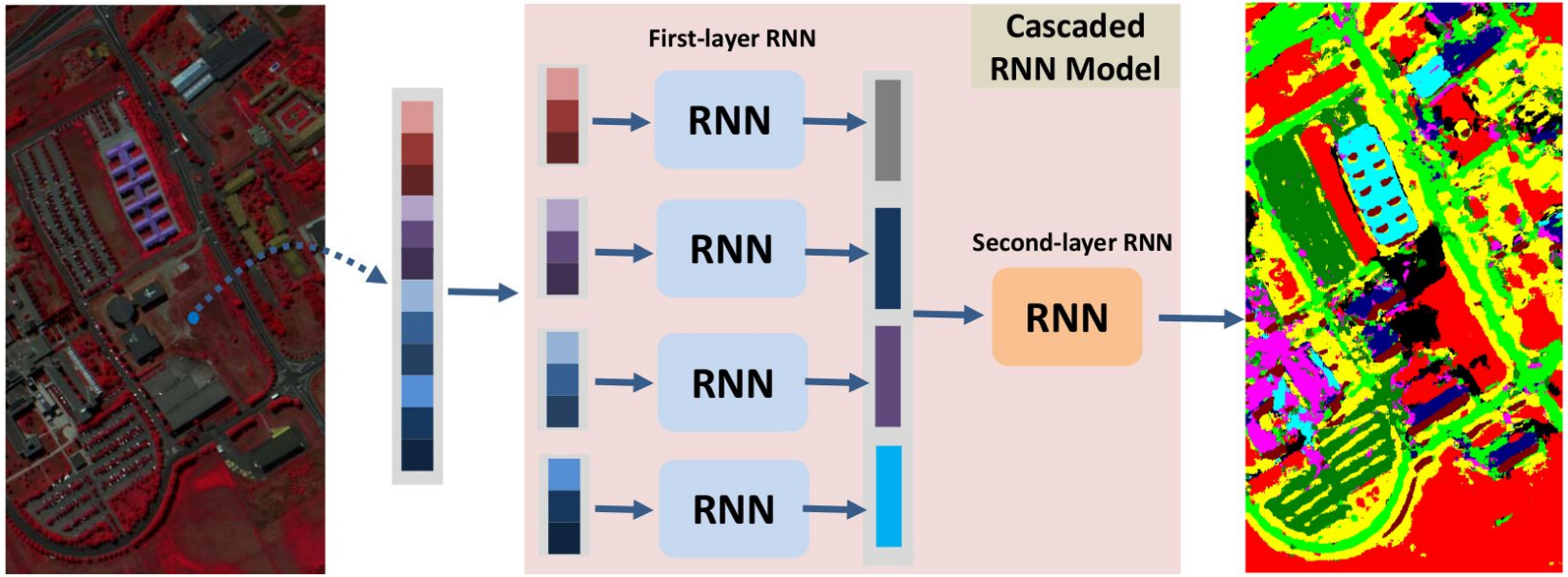


Classification Map for RNN



Cascaded Recurrent Neural Network (CasRNN)

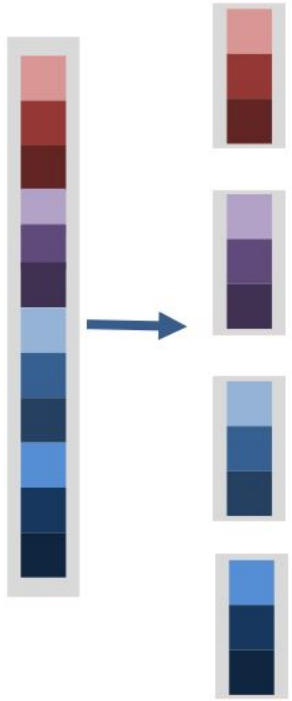
- contains two RNN layers;
- the first layer reduces redundant information;
- the second layer learns complementary information.



Hang, Renlong & Liu, Qingshan & Ghamisi, Pedram. (2019). Cascaded Recurrent Neural Networks for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing. PP. 1-11. 10.1109/TGRS.2019.2899129.

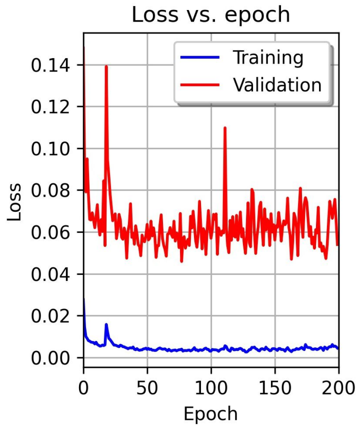
Hyperparameters

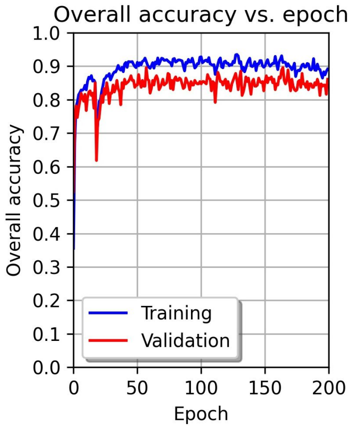
- As before:
  - learning rate;
  - number of epochs;
  - hidden size of RNN;
  - number of layers for RNN;
  - dropout.
- CasRNN specific:
  - sub-sequence number **b**:
    - we divide the spectral sequence **z** into **b** sub-sequences;
    - **z** = (z_1 , z_2 , · · · , z_b), where **z** is a spectral sequence;

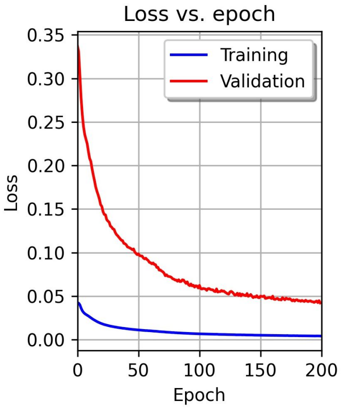


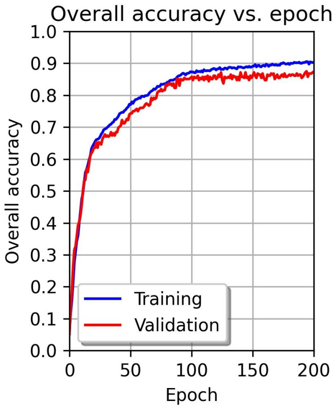
Experiments for CasRNN

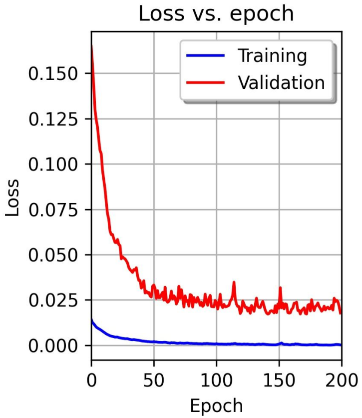
- Learning rate:
  - lr = {0.0001, 0.001, 0.01}
    - 0.0001 bottom left;
    - 0.001 bottom right;
    - 0.01 top right;
  - hidden size 1 = hidden size 2 = 80;
  - num layers 1 = num layers 2 = 2;
  - dropout 1 = dropout 2 = 0.

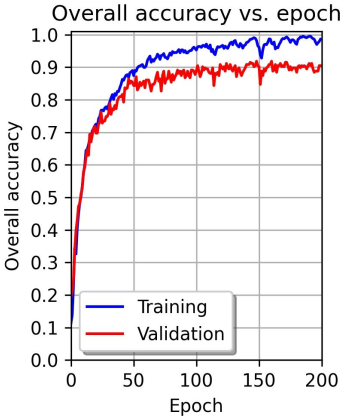






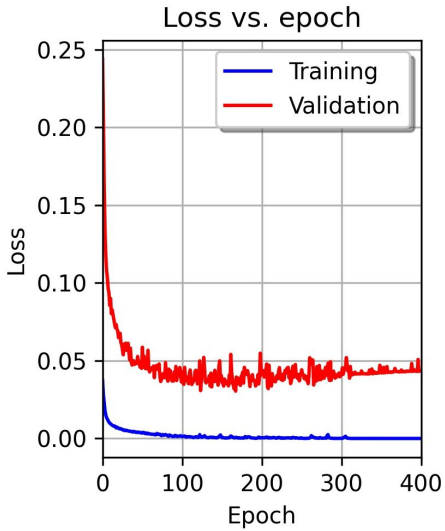


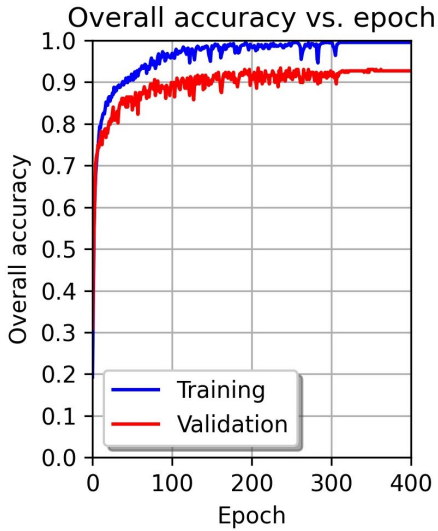




Experiments for CasRNN

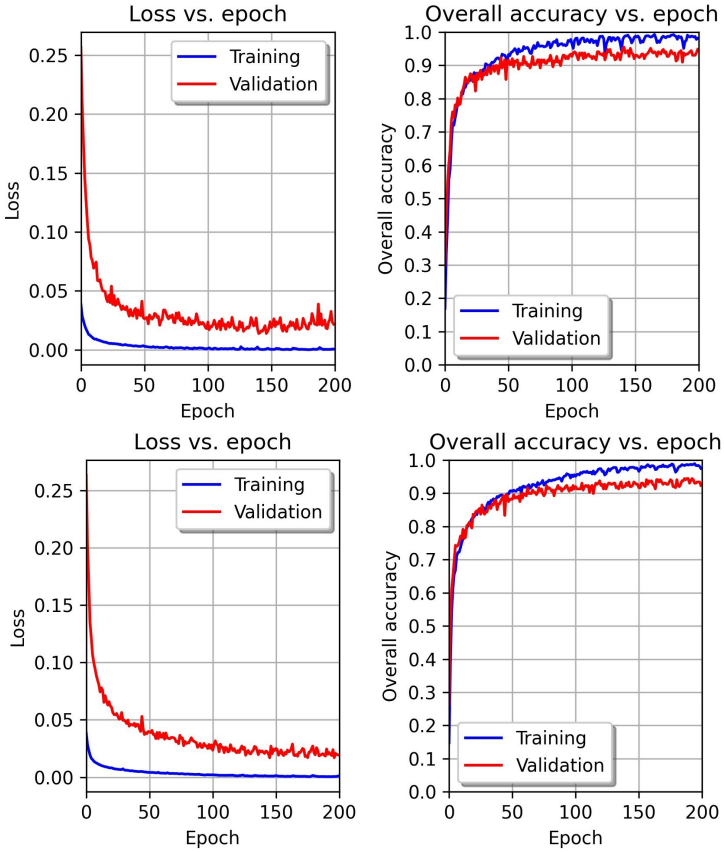
- Number of epochs:
  - lr = 0.001;
  - hidden size 1 = hidden size 2 = 80;
  - num layers 1 = num layers 2 = 2;
  - dropout 1 = dropout 2 = 0.
  - Observation:
    - validation loss starts to grow after 200 epoch ⟹
    - possible overfitting ⟹
    - fix 200 epochs.





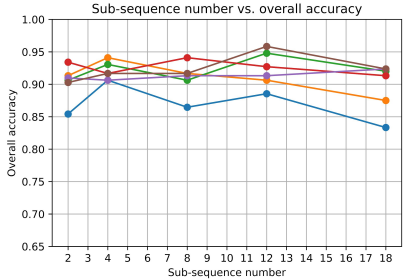
Experiments for CasRNN

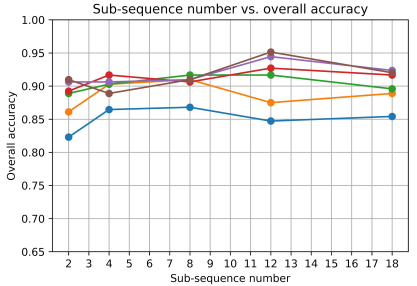
- Number of layers:
  - all combinations of
    - number layers 1 = {1, 2, 3};
    - number layers 2 = {1, 2, 3};
      - (1, 1) bottom;
      - (3, 3) top;
  - lr = 0.001;
  - epoch = 200;
  - hidden size 1 = hidden size 80;
  - dropout 1 = dropout 2 = 0.

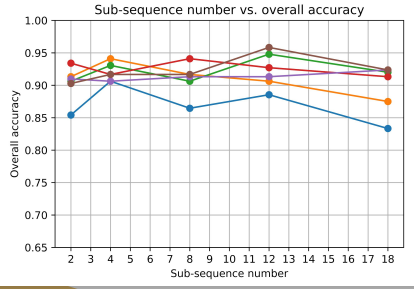


Experiments for CasRNN

- Hidden size and sub-sequence number:
  - all combinations of:
    - hidden size 1 = {16, 32, 64, 128, 256, 384}
    - hidden size 2 = {16, 32, 64, 128, 256, 384}
    - sub-sequence number = {2, 4, 8, 12, 16, 18}
  - lr = 0.001;
  - epoch = 200;
  - num layers 1 = num layers 2 = 1.

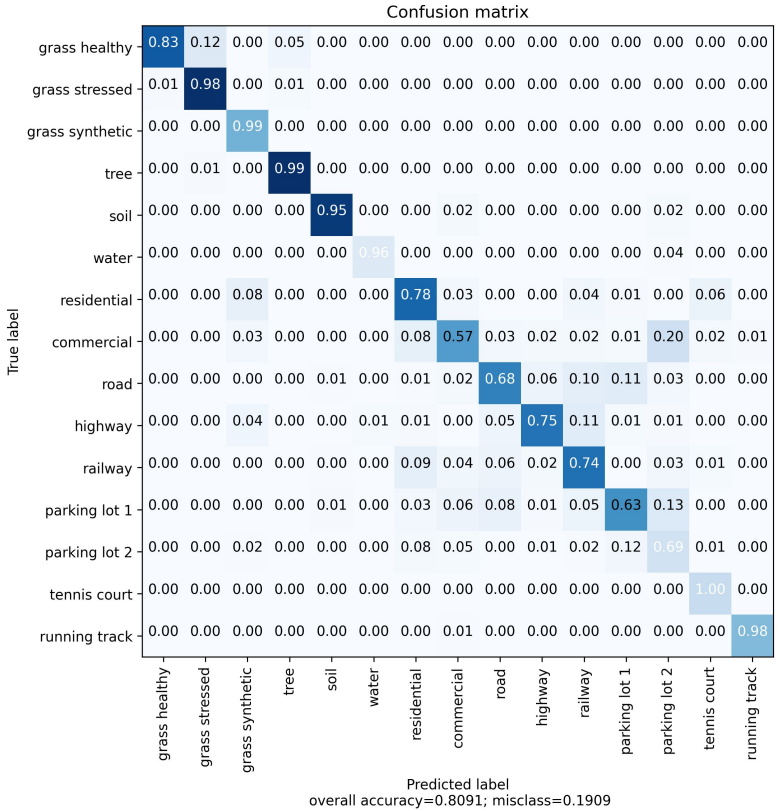




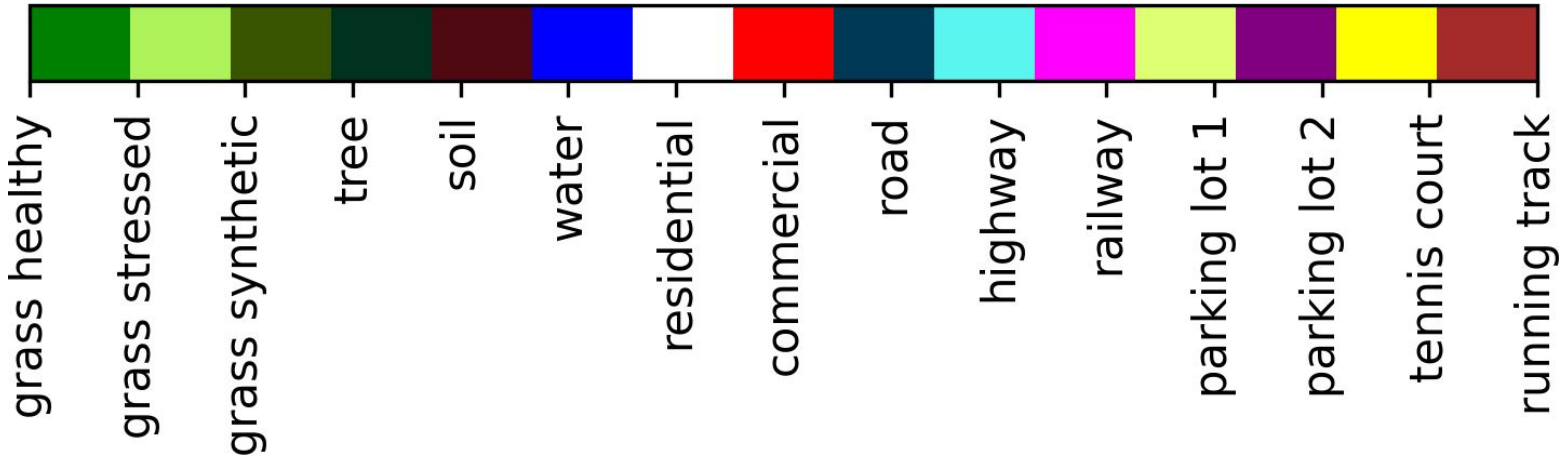


Confusion matrix for CasRNN

- Final parameters:
  - lr = 0.001;
  - epoch = 200;
  - num layers 1 = 1;
  - num layers 2 = 1;
  - sub-sequence number = 12;
  - hidden size 1 = 256;
  - hidden size 2 = 384;
- Best predictions:
  - grass stressed (98%), grass synthetic (99%), water (96%), running track (98%); tennis court (100%)
- Worst predictions:
  - commercial (57%)



Classification Map for CasRNN



Comparison of RNN and CasRNN

- CasRNN outperforms RNN:
  - at 5% for OA, AA and Kappa;
- CasRNN has 37 times more parameters;
- RNN is faster:
  - 4 times at training;
  - 3 times at inference;
- RNN is more efficient (3.5 times).

Parameters	RNN	CasRNN
Overall accuracy, %	76.17	80.8
Average accuracy, %	79.01	83.9
Cohens kappa, %	74.22	79.3
Number trainable parameters	60015	2226063
Training time, sec	68.8	293.6
Inference time, sec	0.14	0.45
Efficiency, pixel / sec	102344	29421

Hang, Renlong & Liu, Qingshan & Ghamisi, Pedram. (2019). Cascaded Recurrent Neural Networks for Hyperspectral Image Classification. IEEE Transactions on Geoscience and Remote Sensing. PP. 1-11. 10.1109/TGRS.2019.2899129.

Comparison of RNN and CasRNN

Conclusion

- CasRNN outperforms RNN in accuracy.
- Confirms the results of the paper on the Houston dataset.
- It is harder to find the best hyperparameters for CasRNN.
- Hyperparameter tuning is crucial topic.

The End

Questions?

Accuracy

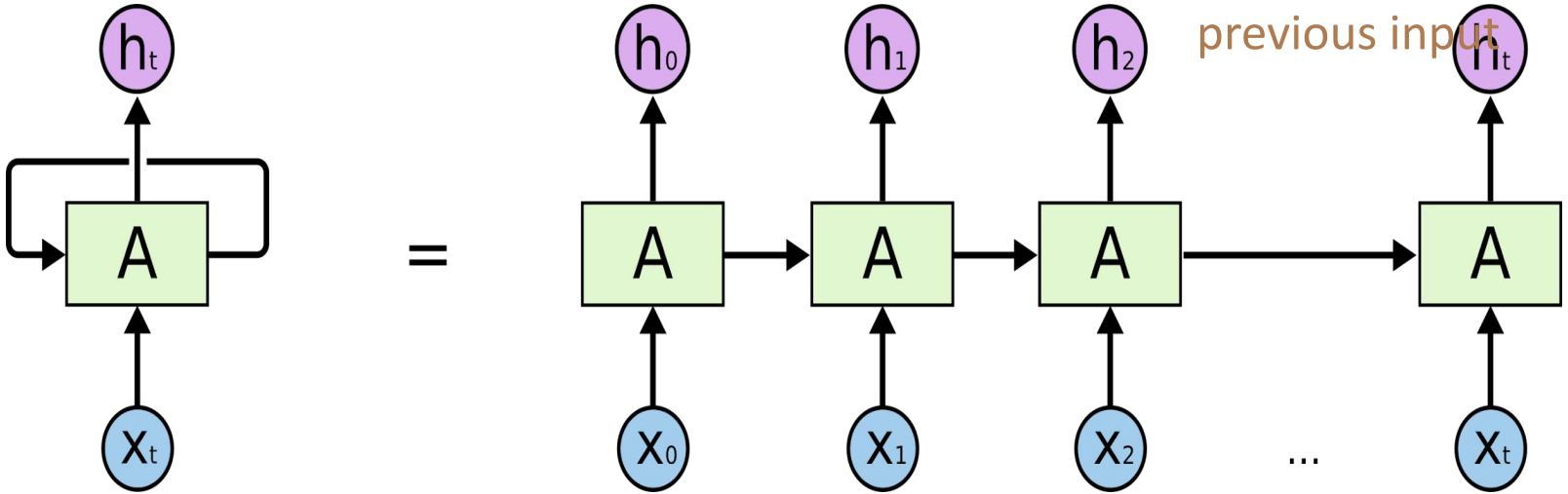
- Overall accuracy defines the ratio between the number of correctly classified pixels to the total number of pixels in the test set;
- Average accuracy refers to the average of accuracies in all classes;
- Kappa is the percentage of agreement corrected by the number of agreements that would be expected purely by chance.

Recurrent Neural Network (RNN)

Input → Hidden → Output

Input + Previous Hidden → Hidden → Output

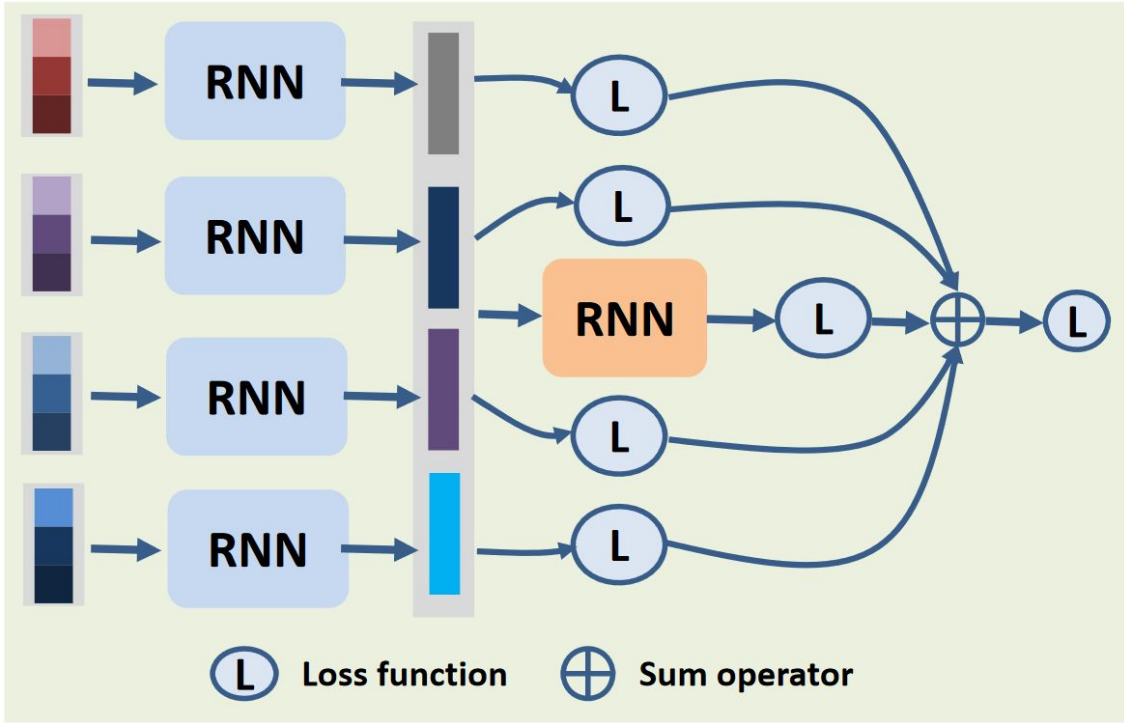
- sequential data
- looks at the current input being presented to it but also the previous input



Improvements of CasRNN

- based on output-level connections (CasRNN-O)
- feed the output of the first and the second layer RNNs into the output layer as a weighted sum

$$\tilde{L} = \frac{1}{l}\sum_{i=1}^{l} w_i^{(1)} L_i^{(1)} + w^{(2)} L^{(2)}$$



Cascaded Recurrent Neural Network (CasRNN)

redundant information among adjacent spectral bands

curse of dimensionality

classification accuracy drops

learn complementary, discriminative features

classification accuracy increases