Deep Fusion Networks for Land Cover Classification

Matthew Pooser Sheng Li
University of Georgia
Athens, GA 30602
{mpooser, sheng.li}@uga.edu

Abstract

Applying computer vision to solve land-based classification tasks have been thoroughly researched, however few papers have been published on combining multimodal imagery data to improve classification performance of land cover classification. We hope to answer whether or not data fusion of multimodal imagery data improves overall classification performance of land cover datasets. We found that one of our deep fusion networks obtained an overall accuracy of 64.29% compared to a network using one dataset. We also found that several deep fusion network strategies work best for specific land cover labels. We hope that further optimization and fine-tuning of the models on different data would confirm these results.

1. Introduction

This project explores the fusion of multimodal imagery data in the hopes of improving classification performance of land cover and eventually land use classification. A few applications of this project include automatic classification of land cover over time, finding potential places to expand infrastructure (commercial & residential), and most importantly aggregating a variety of data from different sources to train models for better land cover classification. Usually, land cover classification uses one dataset. Instead of using a singular dataset, we hope that fusing together different forms of imagery data from satellite surveys would yield higher overall classification compared to using a singular dataset. Therefore, we hope to answer the following question: does data fusion improve classification accuracy of land cover classification? We hope to make the following contributions:

1. The creation of deep fusion CNNs for land cover classification
2. Training these deep fusion CNNs in a Python environment on our university's High Performance Cluster (HPC)

2. Related Work

Recall that this project aims to determine if data fusion improves the classification accuracy of land cover and eventually land use. Land Cover is simply the physical material that the satellite observes, whereas Land Use refers to the main use for a particular patch of land. Thus, some examples of land cover may be water, grass, trees, asphalt, etc. Some examples of land use include recreational, transport, agricultural, residential, and commercial.

In order to observe an increase of classification accuracy for satellite images, we developed a variety of new Convolutional Neural Networks (CNNs) based off of ResNet101, called Deep Fusion Networks. A CNN is simply a filter (referred to as a kernel) that passes over the entire image, multiplying the values of a group of pixels by the values of the kernel to produce a smaller shaped matrix. The shape of the kernel and how many pixels to pass over before performing the next convolution can be determined by researchers at their discretion. Let f denote the input image and h denote the kernel. Let G be the resultant matrix with m and n as rows and columns respectively. The calculation to find the pixels of G are found with the following equation:

$$G[m,n] = (f*h)[m,n] = \Sigma_j \Sigma_k h[j,k] f[m-j, n-k] \quad (1)$$

In the middle of a CNN, there are a variety of operations executed sequentially such as more convolutions and batch normalization, which are repeated until the end of the network. At the end, average pooling and a fully-connected layer similar to that of Multi-Layer Perceptrons whose output is a set of nxp values, where n refers to the number of pictures and p refers to the number of class labels. Therefore, for every image, there will be p values that correspond to the probability for each class label. See Fig. 1 for an example ResNet architecture. Our new networks will be discussed in further details in the Experimental Design section.

While trying to learn more about Land Use/Land Cover and CNNs, we found a paper which discuss combining 2

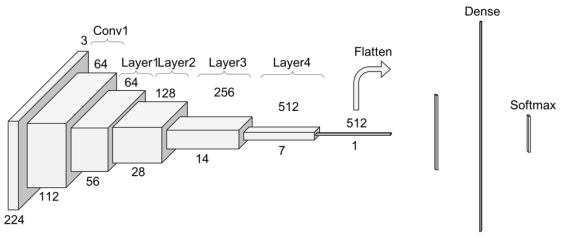

Figure 1. Toy-like example of ResNet architecture

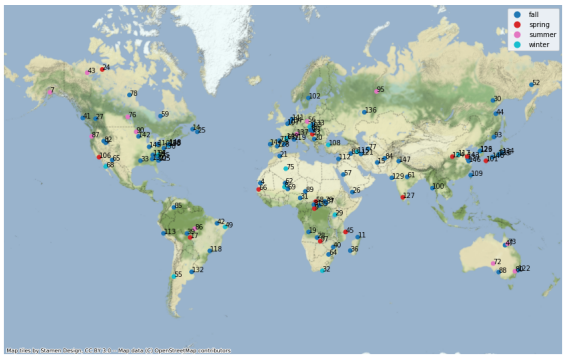

Figure 2. Map of some SEN12MS samples from the SEN12MS dataset

datasets provided by the European Space Agency: Sentinel-1 and Sentinel-2, and performs a basic comparison using Sentinel-1, Sentinel-2, and both of them on some basic deep learning architectures including VGG, DenseNet, and ResNet. This dataset, called SEN12MS, consists of approximately 181,000 triplets of dual-polarization synthetic aperture radar (SAR) image patches (Sentinel-1), multi-spectral image patches (Sentinel-2), and land cover maps [1]. Some applications of Sentinel-1 include marine and land monitoring while applications of Sentinel-2 include monitoring land cover change and agricultural applications (crop monitoring). Each picture is 256 x 256 pixels, and they are sampled from all seasons. See Fig. 2 for an illustration of some examples where the pictures were taken from around the world. Other papers have discussed fusing DSM and UAV images for land cover classification such as [2]. They found that a singular dataset associated with height information (the fused orthomosaic photo and DSM) performed better in most of the discriminative classes. This paper instead focuses on fusing Sentinel-1 (polarized images) and Sentinel-2 (RGB images) to improve land cover classification, and eventually land use classification. At the moment, land use classification is not possible with SEN12MS because the class labels correspond to land cover. Despite this, by fusing both datasets, we hope to find an overall increase in classification accuracy by fusing data in 3 different scenarios: early, middle, and late.

Set	Samples
Train	108,912
Val	53,643
Test	18,106

Table 1. Number of samples used for train, validation, and test sets

Label No.	Type
1	Forest
2	Shrubland
3	Savanna
4	Grassland
5	Wetlands
6	Croplands
7	Urban/Built-up
8	Snow/Ice
9	Barren
10	Water

Table 2. IGBP simplified labels used by SEN12MS

3. Experimental Design

For all our experiments, we created 2 pickle (PKL) files for the training and validation datasets. As for the test set, we used the SEN12MS authors' provided PKL file. See Table 1 for the number of samples used for training, validating, and testing respectively.

Each experiment used mini batches of 25 samples and 100 epochs but slightly differed in learning rate and decay. We used the simplified International Geosphere-Biosphere Programme (IGBP) scheme for our class labels which can be found in Table 2.

3.1. Testing Environment: Sapelo2

At first, we ran a test trial of creating a ResNet50 using a small portion of the training set for Sentinel-1 only (50,000 samples) on a system with an AMD Ryzen 9 3900x and NVIDIA RTX2070 graphics card. It took approximately 3 days for training to complete. Afterward, all training, validating, and testing was executed on the University of Georgia's GACRC HPC: Sapelo2. Sapelo2 has different partitions for different needs, and we primarily focused on using the GPU nodes to make use of PyTorch's CUDA operations. The specific GPU node we used for testing had an Intel Xeon Skylake processor (32 cores and 187GB of RAM) and 1 NVIDIA P100 GPU card. After running our test trial on this specific node on Sapelo2, training/validation finished in around 24 hours. We then ran all of our experiments on Sapelo2. Here is an example of a submission script for running one of the deep fusion networks for training/validating on Sapelo2:

```
#!/bin/bash
#SBATCH --job-name=s1s2_msplit          # Job name (testBowtie2)
#SBATCH --partition=gpu_30d_p           # Partition name (batch, highmem_p, or gpu_p)
#SBATCH --gres=gpu:P100:1               # request 2 GPUs
#SBATCH --ntasks=1                      # Run job in single task, by default using 1 CPU core
 on a single node
#SBATCH --cpus-per-task=5               # CPU core count per task, by default 1 CPU core per
task
#SBATCH --mem=70G                       # Memory per node (4GB); by default using M as unit
#SBATCH --time=25-00:00:00              # Time limit hrs:min:sec or days-hours:minutes:second
s
#SBATCH --export=NONE                   # Do not export any user's explicit environment varia
bles to compute node
#SBATCH --output=log.%j.out             # Standard output log, e.g., log.12345.out
#SBATCH --error=log.%j.err              # Standard error log, e.g., log.12345.err
#SBATCH --mail-user=mpooser@uga.edu     # Where to send mail
#SBATCH --mail-type=ALL                 # Mail events (BEGIN, END, FAIL, ALL)


cd $SLURM_SUBMIT_DIR                    # Change directory to job submission directory (Optio
nal!)

ml Anaconda3/2020.02                    # Load software module and run bowtie2 below
source activate cv                      # load environment

python ./main_train.py --exp_name comb_dfnn1_s1s2_mid --data_dir ../ --label_split_dir ../lab
els_splits --IGBP_simple --threshold 0.1 --model DFNN1 --lr 0.004 --decay 1e-6 --split_networ
ks --use_s1 --use_s2 --fusion_point middle --batch_size 25 --epochs 100
```

Figure 3. Sapelo2 submission script for middle fusion

3.2. Python Implementation of Deep Fusion Networks

Some starter code was provided by the creators of SEN12MS with basic neural network implementation for VGG, ResNet, and DenseNet. In order to facilitate the creation of 3 different deep fusion networks, we adjusted the Python code to handle 2 new scenarios: fusion in the middle and fusion in the end, and used the command line to implement the last scenario: fusion in the beginning. After running a series of basic neural network tests, we found that ResNet101 is the best fit and used that as a baseline for the rest of our experiments.

To handle fusion in the beginning, we simply use the `--use_s1` and `--use_s2` parameters in our cluster submission script. This will then fuse the data for both datasets and feed them as input into a single ResNet101.

As for handling fusion in the middle, a "DFNN1" (Deep Fusion Neural Network) container creates 3 ResNet101 networks: first for Sentinel-1, the second for Sentinel-2, and the third for the fused data. After the 2nd hidden layer, the tensors are averaged together and the forward pass finishes using the third network (now with the fused data).

In order to handle fusion in the end, two ResNet101 networks are created with Sentinel-1 and Sentinel-2 data separately. Right before the activation layer, the logits created by the forward pass are averaged together and fed into the activation function (either $sigmoid$ or $Softmax$).

4. Results

Table 3 shows the results of the Sentinel-1, Sentinel-2, data fusion in the beginning, data fusion in the middle, and data fusion in the end using ResNet101. It shows the Overall Accuracy (OA), Average Accuracy (AA), and Class-wise Accuracy (CWA) for all target classes. OA is calculated by dividing the sum of correctly predicted items by the total items to predict. AA is calculated by dividing the sum of all class-wise accuracies by the number of target classes. See Sec. 5 for why AA may not be a reliable metric. Middle fusion performed the best with respect to OA (marginally higher than Sentinel-2) and AA. Fusion in the end (or late deep fusion) performed the worst, slightly outperformed by Sentinel-1.

ResNet101	OA	AA	CWA									
Test set			1	2	3	4	5	6	7	8	9	10
Sentinel-1	50.92%	87.97%	85.72%	86.31%	68.54%	68.02%	96.77%	81.46%	96.28%	100%	98.09%	98.50%
Sentinel-2	64.26%	91.91%	89.64%	94.54%	83.43%	72.44%	97.17%	87.09%	96%	100%	99.51%	99.26%
Early fusion	62.86%	90.90%	89.37%	92.10%	78.41%	71.25%	97.74%	84.54%	96.75%	100%	99.68%	99.19%
Middle fusion	64.29%	91.30%	89.09%	86.86%	82.12%	76.36%	97.42%	86.98%	95.51%	100%	99.31%	99.35%
Late fusion	49%	87.59%	84.94%	94.79%	75.89%	62.59%	97.14%	72.68%	92.18%	100%	99.84%	95.88%

Table 3. Test set results for all neural networks

Training/Validation Results	Train loss	microAcc	microF1	microF2
Early Fusion	0.08	0.964	0.888	0.881
Middle Fusion	0.083	0.962	0.879	0.868
Late Fusion	0.05	0.871	0.603	0.58

Table 4. Train/validation results for Deep Fusion Networks

With respect to CWA, class 8 (snow/ice) was perfectly classified by all networks although this could have been an issue in the test set or some training error. Looking at the other target classes, Sentinel-2 outperformed the other networks except in a few cases. Looking at the deep fusion networks separately from Sentinel-1 and Sentinel-2, each network performed well in certain target classes. For example, early fusion performed the best for classes 1, 5, and 7 while middle fusion performed the best for classes 3, 4, 6, and 10.

Table 4 shows the training/validation results for the deep fusion networks in terms of train loss, microAcc, microF1, and microF2. A micro-average takes into consideration the contributions of all target classes to compute the average metric. Late fusion performed the worst in terms of microAcc, microF1, and microF2 yet achieved the best train loss out of all deep fusion strategies. Early fusion performed the best in terms of microAcc, microF1, and microF2 but was marginally higher than middle fusion's microAcc and microF1.

5. Discussion

In the results section, we mentioned how AA may not be a reliable metric used to identify the effectiveness of the networks. This is primarily due to how the test set could influence the results of class-wise accuracy. Recall that AA is calculated by summing all the class-wise accuracies and dividing by the number of target classes for every network. Because class 8 (snow/ice) was classified 100% of the time in the test set, all networks' AA is slightly higher than what it would be if we were to drop class 8 from the calculation for AA. A couple reasons why it could be 100% may stem from data leakage and/or a faulty test set. Since other results

had scores lower than 100% for all networks, data leakage may not be to blame.

For the most part, it would seem as though data fusion yields marginal improvements over using one dataset given how middle fusion had an OA of 64.29% compared to Sentinel-2 whose OA was at 64.26%, a difference of 0.03%. However, deep fusion networks may have some niche use cases. Looking at CWA, 60% of the target classes had higher accuracy in deep fusion networks compared to Sentinel-2: classes 5 & 7 (early fusion), classes 4 & 10 (middle fusion), and classes 2 & 9 (late fusion). It may be possible for some researchers to extend upon and improve these small performance gains over using one dataset. One source for the marginal improvements may come from the type of data used. Sentinel-2 is multi-spectral data with 13 bands (visible, near IR, and short wave IR) while Sentinel-1 uses polarized light. Thus, the fusion of different or in fact more datasets may yield better results. Careful consideration would need to be made as execution time and memory consumption would increase from the explosion of parameters required to train bigger deep fusion networks. More fine tuning to hyperparameters as well as network implementation may also be needed to further improve upon the performance increases of data fusion.

It took approximately 5 days to train/validate for early and late; while, it took 7 days for middle fusion. A well-equipped HPC is required to train these networks or there will be insufficient memory for the network to fully be trained. It may be possible to optimize the implementation of deep fusion networks such that the memory consumption and/or execution time may be improved. If this is possible and improvements to classification accuracy are further improved, then deep fusion networks may very well be used in place of using a single dataset.

5.1. Future Work

Some future work could extend deep fusion network research by incorporating code optimization to consume less memory and minimize execution time. Furthermore, training deep fusion networks on different datasets should be used to confirm these performance gains over using a singular dataset. Additionally, ensemble learning using different deep fusion strategies for certain target classes may even further boost classification accuracy. Finally, since the data only has labels for land cover instead of land use, different labels would need to be created to better suit a land use classification task.

6. Conclusion

We found deep fusion networks slightly improve classification accuracy compared to a network using a singular dataset. Middle fusion networks had the highest OA compared to the other networks, but different deep fusion strategies had higher CWA than other networks depending on the target class except in 4 target classes.

References

[1] Schmitt M, Hughes LH, Qiu C, Zhu XX (2019) SEN12MS - a curated dataset of georeferenced multi-spectral Sentinel-1/2 imagery for deep learning and data fusion. In: ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences IV-2/W7: 153-160. 2

[2] Al-Najjar, Husam & Kalantar, Bahareh & Pradhan, Biswajeet & Saeidi, Vahideh & Abdul Halin, Alfian & Ueda, Naonori & Mansor, Shattri. (2019). Remote Sensing Land Cover Classification from Fused DSM and UAV Images Using Convolutional Neural Networks. Remote Sensing. 2019. 1461. 10.3390/rs11121461. 2