Creating Satellite Image Embeddings with Multi-Task Learning

Project Category: Computer Vision

Nandita Naik
nanditan
Department of Computer Science
Stanford University
nanditan@stanford.edu

Bay Foley-Cox
bayfc
Department of Computer Science
Stanford University
bayfc@stanford.edu

1 Introduction

Satellite data is a rich source of information about our planet. Previous analysis has used satellite images for detecting deforestation [1], studying malaria outbreaks [2], and analyzing human impact on the oceans [3]. Yet training computer vision models on satellite data is resource intensive and time consuming. Due to the curse of dimensionality, analyzing satellite images also requires large amounts of labelled data [4]. Despite the rich representation of satellite data, the applications of learning from them are limited because it is difficult to afford the resources necessary to train satellite image-based models. Therefore, reducing the dimensionality of satellite images while preserving their meaning can help researchers efficiently address vital global challenges.

Thus, we focus on the problem of representing satellite images via embeddings. We hypothesize that similar features of satellite imagery are relevant to many prediction tasks. For instance, information about buildings might apply to measuring population density, home prices, and CO_2 emissions. This motivates us to generate embeddings using multi-task learning, which has shown success in other applications [5]. To the best of our knowledge, no prior work has used multi-task learning for embedding satellite images.

Our approach trains a deep neural network to perform multiple regression tasks on a satellite image simultaneously, then uses the output of the last hidden layer as the embedding. We use this trained model to take in a satellite image and produce a lower-dimensional embedding. We evaluate our embeddings based on their performance when used to train linear regression models for unseen tasks. We also demonstrate that our model can transfer its learning to previously unseen tasks, and our results on unseen tasks show an improvement over MOSAIKS, the current state of the art approach for satellite image embeddings [6]. Specifically, for predicting population density, with the same amount of embedding dimensions, we improve MOSAIKS' R^2 score by 16%, and for predicting nightlights, we improve MOSAIKS' R^2 score by 21%.

Our key contributions include: (1) a novel method for generating satellite image embeddings using multi-task learning, (2) an exploration of the embeddings we've generated, and (3) a comparison of our methods and performance to prior work.

2 Related Work

2.1 Satellite Image Embeddings

Previous satellite image embeddings use unsupervised approaches. For instance, Tile2Vec (2019) trains a CNN to embed image tiles such that spatially close tiles are close in the embedding space [7]. One advantage of this approach is that it requires no labels and is task independent. One disadvantage is that it uses a large, computationally-expensive CNN. RegionEncoder (2019) is an unsupervised autoencoder that learns multimodal embeddings from satellite images combined with other data about the region [4]. This approach yields excellent results, outperforming Tile2Vec on some regression tasks. However, their multimodal embeddings require other data in addition to the images, and this data is not available everywhere, making it a less general approach.

MOSAIKS (2021) is a set of task-independent satellite image embeddings that can solve different prediction tasks [6]. To construct the embeddings, the MOSAIKS team first used a random kitchen sink approach where they extracted random convolutional features (RCFs) from a training set. Then to generate component i of their embeddings for an input image, they convolve RCF i over the image and average the results. The benefits of this approach is that is that is highly efficient, and training is label-free. However, the shallow architecture of the model can't create features representing complex functions of the input image, which could be highly beneficial for some tasks.

The MOSAIKS unsupervised approach outperformed prior work on embedding satellite images, including approaches using large CNNs, and their technique is the current state of the art. We compare our embeddings to MOSAIKS' throughout our project, benchmarking our performance against theirs. MOSAIKS makes their dataset publicly

available, which made this project feasible and allows us to make direct comparisons with their approach.

2.2 Multi-Task Learning

Multi-task learning (MTL) solves multiple tasks simultaneously, utilizing inherent connections between related tasks to create models that generalize well [8], [5]. Training on multiple tasks helps avoid over-fitting to a single task [8]. The model also learns to implicitly prioritize the features that the tasks prefer [8]. Based on this reasoning, we decided that multi-task learning could help our embeddings prioritize important features and generalize better to unseen tasks.

Within computer vision, Garcia et al. (2019) used a supervised multi-task learning model trained on four art classification tasks to create visual embeddings for automatic art analysis. This approach achieved high performance compared to baseline approaches [5]. We drew from this paper the idea of applying multi-task learning to embeddings.

3 Dataset and Features

For our dataset, we build off the approach of MOSAIKS in order to make meaningful comparisons [6].

3.1 Geographic Sampling

The MOSAIKS team provides a set of 100,000 geographic coordinates in the continental United States sampled uniformly at random, alongside labels for regression tasks associated with the 1km x 1km square area centered at the coordinates. From this dataset, we use a random subset of 33,085 locations due to cost constraints on acquiring and processing imagery. We use 13,085 locations for the training phase for our multi-task models using an 80-10-10 train-test-validation split. We use 20,000 locations to evaluate our embeddings, also with an 80-10-10 train-test-validation split.

3.2 Imagery

The inputs to our model for each location are satellite images obtained from the Google Static Maps API [9]. Following the approach in MOSAIKS, we obtain 640 x 640 pixel images at zoom level 16, which correspond to land areas of approximately 1 km^2. These images have standard RGB color channels and were geo-rectified and preprocessed to eliminate clouds. This is an example image from our dataset:

Because we finetune models starting from pretrained weights, we must adopt the transformation associated with the original training. In the case of the vision transformer models used in this paper, this consists of reducing the resolution to 242x242 and central cropping to 224x224. The values are then scaled to [0,1] and normalized to have mean [0.485, 0.456, 0.406] and standard deviations [0.229, 0.224, 0.225] across the three color channels [10].

3.3 Prediction Tasks

MOSAIKS provides labels for six regression tasks (see Table 1). We preprocess the labels for each task to have a mean of zero and standard deviation one when they are fed into our multitask model, in order to ensure that tasks with higher numerical values are not weighted more heavily. For linear regression, we predict the labels directly, following the approach of the MOSAIKS paper.

Table 1: MOSAIKS Tasks

Task	**Unit**
Forest Cover	forest percentage
Elevation	meters
Population Density	log(1 + people/km^2)
Nighttime Lights	log(1 + nanoWatts/cm^2 / sr)
Income	$ per household
Road Length	meters

In order to compete as fairly as possible against the unsupervised approach in MOSAIKS, we randomly selected four tasks (Elevation, Road Length, Income, Nighttime Lights), as training tasks for use in training our multi-task model. We reserve the remaining tasks for evaluating our embeddings against MOSAIKS.

4 Methods

4.1 High-level Overview

We divide our project into three principal stages as shown in figure 1.

In **Stage 1: Train Multi-Task Model**, we fine-tune a vision transformer on multiple regression tasks [11]. Our model takes in images and outputs predictions on our training tasks. We modify the architecture of the vision transformer to generate embeddings, as discussed below.

In **Stage 2: Embed Images**, we generate image embeddings using our fine-tuned vision transformer. We do this by running inference on an image and extracting the final hidden layer output as our embedding. We embed a set of evaluation images which is distinct from our training images.

In **Stage 3: Predict New Tasks**, we divide our embeddings into a train, test, and validation set as described in the dataset section. We then train a linear regression model using our embeddings to predict an evaluation task's labels. Then, we test our linear regression model on our evaluation task test labels.

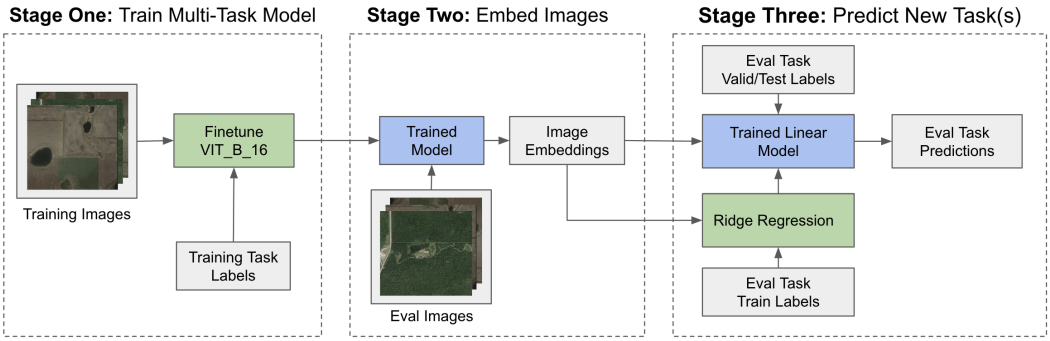


Figure 1: An overview of our embeddings pipeline

4.2 Task formulation

We formulate our multi-task learning problem as follows. Given T learning tasks, with n data points, where the dataset for the task t is $\{x^t, y^t\}_{i=1}^n$, we want to optimize

$$\arg\min_w \sum_{t=1}^{T} \sum_{i=1}^{n} \lambda^t \ell_t(f(x_i^t; w^t), y_i^t),$$

where λ^t indicates the task weight for the t-th task, ℓ_t represents the loss function for the t-th task, and $f(x_i^t; w^t)$ represents the prediction for the i-th x data point for task t, where f is parameterized by w [5]. In this paper, we take $\lambda_t = 1$ and leave task weighting for future work. We also use MSE as our loss function.

4.3 Model Architecture

To generate embeddings, we fine-tuned a vision transformer on our dataset. We use vision transformers, as these models have demonstrated state of the art performance on many computer vision tasks and have been shown to perform well on small and medium-sized datasets when they are pre-trained on ImageNet [11].

We specifically use the vision transformer architecture VIT_B_16 as implemented in PyTorch and trained on the IMAGENET1K_V1 dataset [10]. We use the smallest available architecture due to computation constraints and our relatively small dataset.

The idea of a transformer first originated in NLP. It makes use of **attention**, a mechanism where the model weights different sections of the input given their learned relevance [12]. Importantly, transformers process the whole input at once. This allows for more parallelization and faster training times. A vision transformer (ViT) is a transformer adapted for use on an image [11]. This architecture extracts patches of size 16×16 from the image, then feeds those patches through a standard transformer encoder.

We modify the existing architecture slightly to create embeddings. To generate an embedding of dimension d, we replace the final layer with a fully connected layer, which takes in the same number of inputs as the original final layer, and has output dimension d. Then, we add a ReLU layer, the result of which is our embedding. Finally, we add another fully connected layer, which takes in the d outputs of the ReLU layer and outputs a feature per regression task.

4.4 Baseline method

We used the pre-trained vision transformer VIT_L_16, without any fine-tuning on predicting tasks from satellite images, to establish a baseline performance. We use the larger model for this task as it results in more output features and thus a higher dimensional embedding.

4.5 Evaluation Metrics

Our goal is to measure how well the embeddings perform on predicting a previously unseen task. We use performance on the linear regression tasks as a proxy for evaluating our generated embeddings. We use R^2 error, computed as $1 - \frac{\sum_i (y_i - h_\theta(x_i))^2}{\sum_i (y_i - \tilde{y})^2}$, where our data and labels are $\{x_i, y_i\}_{i=1}^n$, our hypothesis is h_θ, and $\tilde{y} = \frac{1}{n}\sum_{i=1}^n y_i$.

5 Results and Discussion

5.1 Multi-Task Embeddings Results

We used Adam [13] optimization and weight decay to reduce overfitting. For hyperparameter selection, we hand-tuned the model and found that a learning rate value of 0.00001, a weight decay value of 0.01, and a batch size of 64 gave the best results. We train for five epochs, after which we observe the validation MSE ceases to decline.

Because we have far fewer training examples for linear regression than MOSAIKS does ($16,000$ vs $60,000$ images) we focus on training lower 1024 dimensional embeddings than the 8192 dimensional embeddings that MOSAIKS found to be optimal for their approach. However, we do evaluate both our embeddings and MOSAIKS' with 4096 dimensions.

In setting the ridge regression parameter λ, we hyperparameter tune to optimize MOSAIKS' performance on the validation set and found that the optimal value for MOSAIKS at both dims 1024 and 4096 was 2. We then use this hyperparameter for all training, making it strictly harder for our results to compare favorably with MOSAIKS.

The results for our embeddings (MTL w/All Four) and MOSAIKS are summarized in Figure X. We observe that our primary approach performs better than MOSAIKS and our baseline model. Table 2 contains plots of our predictions alongside MOSAIKS.

The baseline model is surprisingly strong, outperforming MOSAIKS, and approaching our performance. This is likely the result of it using a larger vision transformer than we were able to finetune. We believe that finetuning on this architecture could yield better results in the future.

5.2 Ablation Study Results

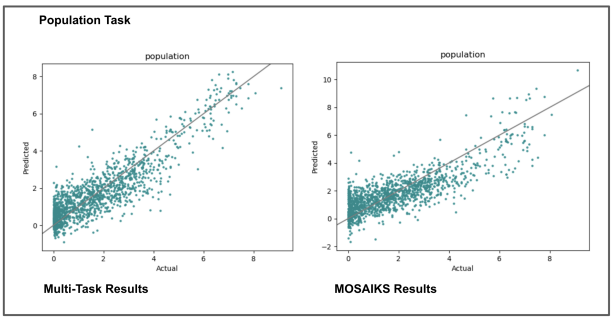
What contribution is each task making to the final trained model? We conducted an ablation study to measure the effect of each train task. To do this, we trained our multi-task model on four train tasks, then removed each task one by one, and evaluated performance.

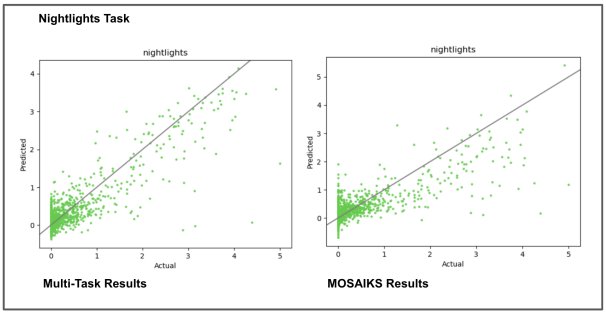
Table 2: R^2 on test tasks for various embeddings.

Tasks	Dim	Population R^2	Nightlights R^2
MTL w/All Four	1024	0.786	0.775
MTL w/o Elevation	1024	0.769	0.774
MTL w/o Income	1024	0.765	0.759
MTL w/o Roadlength	1024	0.768	0.749
MTL w/o Treecover	1024	0.766	0.778
MOSAIKS	1024	0.673	0.638
ViT Baseline	1024	0.739	0.743
MOSAIKS	4096	0.698	0.668
MTL w/All Four	4096	0.736	0.755

Our results show an improvement over MOSAIKS in all models, with the four-tasks model achieving an improvement of 16% on nightlights, and 21% on population. The model trained on all four tasks had the best performance, suggesting that our multi-task approach is successful. Our model also does not appear to overly rely on any single task, since removing a task did not cause our performance to dip by a significant amount. Interestingly, we see that removing elevation had a significant effect on the population task, while not affecting the nightlights task too much. Removing roads, however, caused the most significant dip in both our nightlights and population performance. This makes sense intuitively as an area with more roads is also more likely to be urban, which affects population and light pollution.

In the following plots, we compare the predictions from our best-performing model (MTL with all tasks, dim 1024) against MOSAIKS' with dim 1024. We plot $y = x$, for reference. The MOSAIKS results are noisier and contain more outliers.





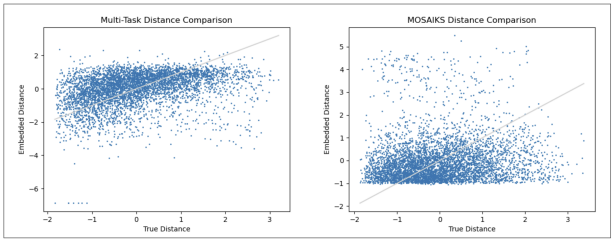
5.3 Embedding Exploration Results

What did the embeddings learn? We conducted experiments to explore the latent space created by our embeddings. We also conduct the same experiments on MOSAIKS' embeddings, contributing analysis which was not present in the original paper [6].

5.3.1 Distance in embeddings, distance in space

Tile2Vec [7] assumes that satellite image tiles that are close geographically will also encode similar semantic meaning. Is this assumption reflected in our embeddings? In MOSAIKS'? To explore these questions, we randomly sample 5,000 pairs of locations and plot their geographic distance against their distance in the embedding space (using the 4096 dimensional variants). We scale the data so that both geographic and embedded distance have mean zero and standard deviation one.

We find the correlation between real distance and embedded distance is weak but positive. The correlation coefficient is 0.313 for our Multi-Task approach and 0.085 for MOSAIKS. These results are not surprising, since there are disparate areas of the United States which appear similar (eg. urban areas) and nearby areas that appear different (eg. parks in cities). We plot the results of this experiment below:



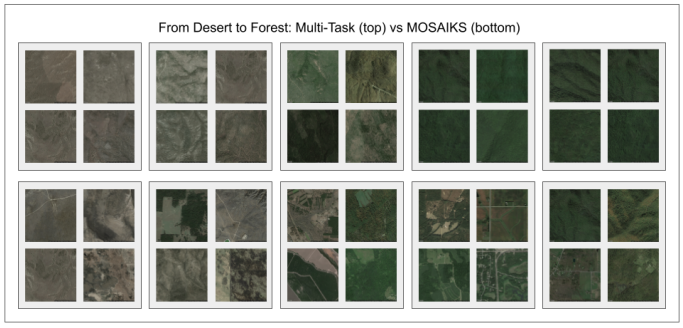
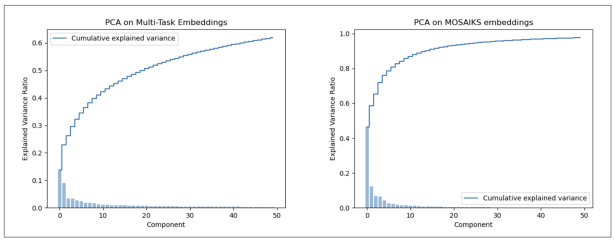


Figure 2: Interpolation plots.

5.3.2 PCA on embeddings

We wanted to explore the extent to which embeddings could be reduced in dimension while retaining information. To this end, we apply PCA to both our embeddings and MOSAIKS'. We ran PCA taking the top 50 principal components and plot the variance explained by the i-th principal component.



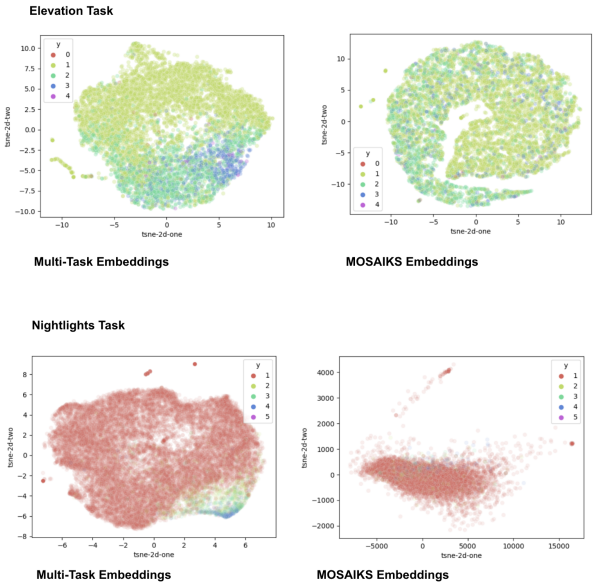
We observe that MOSAIKS has much more variance explained by its top few principal components, indicating its embeddings may contain less information. This could be because its unsupervised approach does not penalize including components highly correlated with existing components. In contrast, our supervised embeddings have a loss function which penalizes the model for having useless features. This distinction could help explain why MOSAIKS performs worse than our embeddings.

5.3.3 Moving through the embedding space

We replicate the **latent space exploration** used by Tile2Vec [7] to explore our embeddings. We sample five uniformly spaced points on the line between the embeddings of two images and find the four nearest neighbors. Figure 2 shows the results of interpolating between a desert image and a forest image in this way. Qualitatively, MOSAIKS has a smoother transition in hue, but our clusters are more meaningfully similar.

5.3.4 t-SNE Visualization

To visualize the embeddings, we used **t-SNE** [14] which uses nonlinear dimensionality reduction to visualize high-dimensional data. We visualized the embeddings trained on all four train tasks, with dimension 1024.



In our t-SNE plots, coloring is based on binning labels for elevation (a train task) and nightlights (a test task). We observe our embeddings differentiate locations based on tasks more distinctly than MOSAIKS'. It is promising that this occurs for nightlights, suggesting the structure of our embeddings generalize to unseen tasks.

6 Conclusion/Future Work

In conclusion, we developed a multi-task learning based method for creating satellite image embeddings that uses knowledge gained from multiple tasks to generalize well to previously unseen tasks. Our methods show improvements of 16% and 21% over MOSAIKS on the test tasks of population density and nightlights density. We also demonstrate that our embeddings preserve meaning.

In future work, we'd like to use more data to better train and evaluate our embeddings. We also want to explore satellite embeddings beyond solving regression tasks. For instance, could studying differences in the embedding of the same location over time give insight into changes on our planet? Through embedding satellite images, we hope to work towards a greater understanding of our planet.

7 Contributions

Bay wrote the data loader and implemented an initial pipeline. Nandita experimented with a variety of architectures and worked to tune hyperparameters, and ran the ablation experiments. We both worked together to design and implement a variety of experiments to explore our embeddings and write up and interpret all of our results.

References

[1] Nor Kumalasari, Caecar Pratiwi, Yunendah Nur Fu'adah, and Edwar Edwar. Early detection of deforestation through satellite land geospatial images based on cnn architecture. 2021.

[2] Thibault Catry, A. Pottier, Renaud Marti, Zhichao Li, Emmanuel Roux, Vincent Herbreteau, Morgan Mangeas, Laurent Demagistri, Helen Da Costa Gurgel, and Nadine Dessay. Contributions of the combination of optical and radar satellite images in the study of vector-borne diseases: the case of malaria at the border between french guiana and brazil. 2018.

[3] Sayed Ishaq Deliry, Zehra Yiğit Avdan, Nghi Tan Do, and Uğur Avdan. Assessment of human-induced environmental disaster in the aral sea using landsat satellite images. *Environmental Earth Sciences*, 79, 2020.

[4] Porter Jenkins, Ahmad Farag, Suhang Wang, and Zhenhui Li. Unsupervised representation learning of spatial data via multimodal embedding. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 1993–2002, New York, NY, USA, 2019. Association for Computing Machinery.

[5] Noa Garcia, Benjamin Renoust, and Uta Nakashima. Context-aware embeddings for automatic art analysis. Proceedings of the 2019 ACM International Conference on Multimedia Retrieval, 2019.

[6] Esther Rolf, Jonathan Proctor, Tamma Carleton, Ian Bolliger, Vaishaal Shankar, Miyaba Isahara, Benjamin Recht, and Solomon Hsiang. A generalizable and accessible approach to machine learning with global satellite imagery. volume 12. Nature, 2021.

[7] Neal Jean, Sherrie Wang, Anshul Samar, George Azzari, David Lobell, and Stefano Ermon. Tile2vec: Unsupervised representation learning for spatially distributed data. volume 33, pages 3967–3974, 07 2019.

[8] Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv: 1706.05098*, 2017.

[9] Google Developers. Maps static API. `https://developers.google.com/maps/documentation/maps-static/overview`. Accessed: 2022-12-7.

[10] VIT_B_32. `https://pytorch.org/vision/stable/models/generated/torchvision.models.vit_b_32.html#torchvision.models.vit_b_32`. Accessed: 2022-12-7.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Laurens van der Maaten and Geoff Hinton. Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research*, 2008.