A Fully Convolutional M2 Variational Autoencoder for Semi-Supervised Semantic Segmentation

Judah Zammit
Department of Computer Science
University of Manitoba
zammitj3@myumanitoba.ca

Abstract

We adapted the M2 variational autoencoder (M2-VAE) for use as a semi-supervised, semantic segmentation model. This is achieved by making the model fully convolutional as well as changing the target distribution from a single Categorical distribution to a Multivariate Bernoulli distribution for each pixel of an image. Gumbel-Softmax sampling is applied to this new target distribution to remove the need of summing over all possible values for target. We achieved a 4% increase of the intersection over union (IOU) coefficient on the Pascal VOC dataset over the fully supervised baseline which achieved a 54% IOU coefficient. This is followed by a discussion on further improvements that could be made to the model.

1 Introduction

To train a supervised machine learning model it is usually necessary to have a dataset that has been hand-labeled by a human. In the field of semantic segmentation, this is an exceptionally time-consuming task. This is for two reasons. One, semantic segmentation makes heavy use of deep convolutional neural networks and these require a substantial amount of data to be effective. And two, to label a photo, it is necessary to assign each of the pixels to one of the target classes.

Semi-supervised models are a way of alleviating this burden by allowing you to make use of both labeled data and unlabeled data. Intuitively, they achieve this by learning both from the structure of the data as well as how the data relates to the target. That is, whatever dependency is found between the data and the target must also explain the behavior of the data in the absence of the target.

Many of the semi-supervised models present in the field of semantic segmentation follow the basic structure of pseudo-labeling. Pseudo-labeling, also known as self-training, is where a model is trained on some labeled data and then applied to some unlabeled data. Its highly confident predictions are then used as labels and the model is retrained with them. This approach is highly heuristic and prone to reinforcing faulty predictions.

Many other semi-supervised models, such as [1], make use of weak labels such as image level labels or bounding boxes and thus are not fully semi-supervised.

Graphical model approaches, such as the M2-VAE[2] and the Skip Deep Generative Model[3], have obtained sizable increases in performance when the target distribution is Categorical. In this paper, we will consider the former.

Conceptually, the M2-VAE can be thought of as an encoder and decoder style model where the encoder takes the original data and outputs some latent variable as well as a prediction for our target. The decoder takes the latent variable and the prediction and attempts to reconstruct the original data. When the target is present, the decoder instead takes the ground truth for the target as input. The model is then trained to reconstruct the original data as accurately as possible. This effectively forces the model to learn to make predictions for the target that are useful for both classification and reconstruction.

The M2-VAE in its original form is only designed to predict an image level label as the target. This is in contrast with the target of a semantic segmentation model which predicts pixel level labels as its target. As noted in [2], the M2-VAE theoretically can be adapted to this new domain. This, however, comes with several unique challenges.

In the original M2-VAE, the target is exactly summed over each of its possible values. This is acceptable when the possible values are relatively small. When the number of possible values becomes quite large, such as for the target of semantic segmentation, this quickly becomes intractable. This is remedied, as suggested in [3], by instead, sampling from the target's distribution using Gumbel-Softmax sampling[4].

The original M2–VAE also makes the simplifying assumption that the target's marginal distribution is Balanced Categorical. Despite it being much farther from the truth in this context, we make a similar assumption here. We assume that each of the possible pixel-wise masks are equally likely to occur. This is obviously not true since, intuitively, we know that a pixel-wise mask close to random noise is much less likely to occur in reality, then, say, a dog-shaped one.

Convolutional networks are the de facto standard in any computer vision task including semantic segmentation. Because of this, instead of using multilayer perceptrons as our encoder and decoder models, we used a modified DeepLabv3+[5] with a MobileNet[6] as its backbone.

2 The M2 Variational Autoencoder

2.1 The Objective

Suppose we are given the following two sets of data: $\mathbb{UN} = \{(x^{(1)}), (x^{(2)}), ..., (x^{(n)})\}$, $\mathbb{LAB} = \{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})\}$. We assume that the data from both was generated by the i.i.d sampling process:

$$z^{(i)} \sim p(z), \;\; y^{(i)} \sim p(y), \;\; x^{(i)} \sim p_\theta(x|z^{(i)}, y^{(i)}). \tag{1}$$

Where $p(z)$ and $p(y)$ are assumed to be independent, $p(z)$ is assumed to be unit gaussian and $p_\theta(\bullet)$ is some distribution parameterized by theta (illustrated in Figure 1).

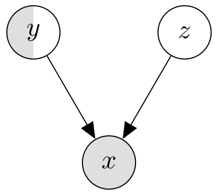


Figure 1: A Graphical Representation of the Generative Process Explaining the Data

Ultimatly, we would like to use this graphical model to find a good prediciton for y given some x. This can be done by finding the values for theta that best explains the given data and then inferring the posterior $p_\theta(y|x)$ from that.

This is intractable for two reasons. To find the optimal values for theta one would have to optimise

$$\begin{gathered} logf(\theta; \mathbb{LAB}, \mathbb{UN}) = \\ \sum_{\mathbb{LAB}} log \int_z p(z)p(y)p_\theta(x|z, y)dz + \\ \sum_{\mathbb{UN}} log \int_z \int_y p(z)p(y)p_\theta(x|z, y)dydz. \end{gathered} \tag{2}$$

The presence of latent variables–that is, the presence of one or more integrals inside of the logarithm–makes this intractable. After finding a suitable theta, we would then be left to calculate the posterior,

$$p_\theta(y|x) = \frac{\int_z p(z)p(y)p_\theta(x|z,y)dz}{\int_y \int_z p(z)p(y)p_\theta(x|z,y)dzdy}, \tag{3}$$

which is also intractable for any sufficiently complex model, including our own.

Instead, we introduce the variational approximation of $p_\theta(z,y|x),\ q_\phi(z,y|x)$. We can then maximize the likelihood of the data by maximizing the following lower bound on the log likelihood of $p_\theta(x)$ and $p_\theta(x,y)$ for each data point. The latter for when y is observed and the former for when it is not. This has the added benefit of minimising the Kullback-Leibler divergence between $q_\phi(y,z|x)$ and $p(y,z|x)$, allowing us to use $q_\phi(y|x)$ as a classifier.

$$\begin{aligned} log p_\theta(x) &\geq \mathbb{E}_{q_\phi(y,z|x)} log \left[\frac{p_\theta(x,y,z)}{q_\phi(y,z|x)}\right] \\ &= \mathbb{E}_{q_\phi(y,z|x)} log \left[\frac{p_\theta(x|y,z)p(y)p(z)}{q_\phi(y|x)q_\phi(z|x,y)}\right] = -\mathbb{U}(x). \end{aligned} \tag{4}$$

$$\begin{aligned} log p_\theta(x,y) &\geq \mathbb{E}_{q_\phi(z|x,y)} \left[log \frac{p_\theta(x,y,z)}{q_\phi(z|x,y)}\right] \\ &= \mathbb{E}_{q_\phi(z|x,y)} \left[log \frac{p_\theta(x|y,z)p(y)p_\theta(z)}{q_\phi(z|x,y)}\right] = -\mathbb{L}(x,y). \end{aligned} \tag{5}$$

We then have

$$\mathbb{J}(\theta,\phi;\mathbb{LAB},\mathbb{UN}) = \sum_{\mathbb{LAB}} \mathbb{L}(x^{(i)},y^{(i)}) + \sum_{\mathbb{UN}} \mathbb{U}(x^{(i)}) \tag{6}$$

as an objective to minimize.

Observe that $q_\phi(y|x)$ appears only in the lower bound for unlabeled data points. This is disadvantageous since our goal is to create an effective classifier. To remedy this, we introduce a modified version of (5),

$$\mathbb{J}(\theta,\phi;\mathbb{LAB},\mathbb{UN})^\alpha = \sum_{\mathbb{LAB}} \mathbb{L}(x^{(i)},y^{(i)}) + \sum_{\mathbb{LAB}} \alpha \mathbb{E}\left[-log q_\phi(y^{(i)}|x^{(i)})\right] + \sum_{\mathbb{UN}} \mathbb{U}(x^{(i)}) \tag{7}$$

where α is a hyperparameter. See [2] for more details on how this is derived.

In our experiments, the various functions appearing in the objective were chosen to be

$$\begin{aligned} p(y) &= Cat(y|\pi), \\ p(z) &= \mathcal{N}(z|\mathbf{0},\mathbf{I}), \\ p_\theta(x|y,z) &= Bern(x|MLP_\theta(y,z)), \\ q_\phi(y|x) &= Cat(y|MLP_\phi(x)), \\ q_\phi(z|x,y) &= \mathcal{N}(z|MLP_\phi(x,y),MLP_\phi(x,y)), \end{aligned} \tag{8}$$

where MLP denotes multilayer perceptron.

2.2 Optimisation

We would like to optimize (7) using a standard gradient descent algorithm. To do so will require the calculation of $\nabla\mathbb{J}(\theta,\phi;\mathbb{LAB},\mathbb{UN})^\alpha$.

To ensure tractiblity, we make the following approximations in the above's calculation:

$$\begin{aligned} \nabla - \mathbb{U}(y) &= \nabla \int_z \sum_y q_\phi(y,z|x) log \left[\frac{p_\theta(x|y,z)p(y)p(z)}{q_\phi(y|x)q_\phi(z|x,y)}\right] dz \\ &\approx \nabla \sum_y q_\phi(y|x) log \left[\frac{p_\theta(x|y,z_i)p(y)p(z_i)}{q_\phi(y|x)q_\phi(z_i|x,y)}\right]. \end{aligned} \tag{9}$$

$$\nabla - \mathbb{L}(x,y) \approx \nabla log \left[\frac{p_\theta(x|y,z_i)p(y)p_\theta(z_i)}{q_\phi(z_i|x,y)}\right] \tag{10}$$

, where $z_i \sim q_\phi(z|x, y)$.

The reparameterization trick[2][7] is used during sampling to maintain differentiability.

3 Adaptation of the M2 Variational Autoencoder to a Bernoulli Target

Suppose we wish to adapt the model from the previous section to a domain where y is a vector of pixel level labels instead of a single image level label. In the above, the Categorical distribution is used to model y. Here a Categorical distribution for each element in y–that is, a Categorical distribution for each pixel–could be used. In our experiments, a Multivariate Bernoulli distribution was used instead, for sake of simplicity.

Furthermore, we must adapt our assumption that $y's$ true marginal distribution is Balanced Categorical. To this end, we instead assume that every possible value for y is equally likely. That is, each pixel map has a probability of $\frac{1}{2^{W \times H \times Y}}$ of occurring. Where W is the width of the photo, H is the height of the photo, C is the number of channels of the photo, and Y is the number of classes each pixel can be classified as.

We are then faced with another problem. In (9) we still must sum over y. This is problematic because the number of possible values for y is no longer just the number of classes, but now $2^{W \times H \times Y}$ and thus it is no longer so easy to calculate. As noted in [3], we can estimate the sum over y in the same way that we estimated the integral over z, by taking a sample from it. In the case of z, the reparametrization trick is a way of doing this without losing differentiability. Such a clean trick unfortunately does not exist for discrete distributions such as $y's$. Instead we use Gumbel-Softmax sampling[4].

Then (9) becomes

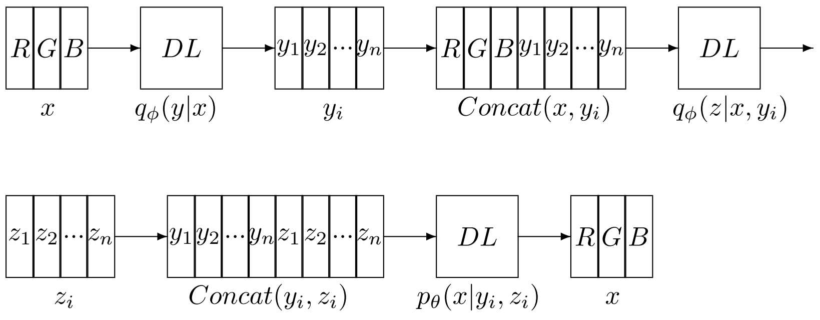
$$\nabla - \mathbb{U} \approx \nabla log \left[\frac{p_\theta(x|y_i, z_i)p(y_i)p(z_i)}{q_\phi(y_i|x)q_\phi(z_i|x, y_i)} \right], \tag{11}$$

,where $y_i \sim q_\phi(y|x)$ and $z_i \sim q_\phi(z, |y_i, x)$.

The functions in our objective are specified as

$$\begin{aligned} p(y) &= Bern(y|\pi), \\ p(z) &= \mathcal{N}(z|\mathbf{0}, \mathbf{I}), \\ p_\theta(x|y, z) &= Bern(x|DL_\theta(Concat(y, z))), \\ q_\phi(y|x) &= Bern(y|DL_\phi(x)), \\ q_\phi(z|y, x) &= \mathcal{N}(z|DL_\phi(Concat(y, x)), DL_\phi(Concat(y, x))), \end{aligned} \tag{12}$$

where the $Concat(\bullet)$ function concatenates the predicted z and y and DL denotes DeepLabv3+ (illustrated in figure 2.)



R G B
x
DL
$q_\phi(y|x)$
y_1 y_2 $\cdots$ y_n
y_i
R G B y_1 y_2 $\cdots$ y_n
$Concat(x, y_i)$
DL
$q_\phi(z|x, y_i)$
z_1 z_2 $\cdots$ z_n
z_i
y_1 y_2 $\cdots$ y_n z_1 z_2 $\cdots$ z_n
$Concat(y_i, z_i)$
DL
$p_\theta(x|y_i, z_i)$
R G B
x

Figure 2: A Depiction of Our Modified M2-VAE

4 Experiments

In our experiments MNIST[8] was used for the original M2-VAE and Pascal VOC[9] was used for the modified one.

In (8) and (12) we very quietly made the assumption that x follows a Bernoulli distribution. Both of the datasets used in our experiments were images so one might expect a Gaussian distribution to be used since pixel intensity is continuous. While not at all theoretically incompatible with either model, using a Gaussian distribution tends to cause training to become unstable[3]. Therefore, we treat pixel intensity as the probability that that pixel is at maximum brightness. Each image can then be viewed as a Bernoulli distribution which we then sample from at each iteration to result in binarized data that can be modeled by a Bernoulli distribution.

Due to MNIST being made of grayscale images, this does not result in much of a loss of information. The same cannot be said about Pascal VOC, which is a dataset of RGB images and thus is affected much more by this binarization.

To simulate a semi-supervised setting, we remove all the 60,000 labels from MNIST's training set except for those of ten images of each class. The 10,000 validation images are left untouched. For Pascal VOC, we simply use the full dataset of 14,000 images as our unlabeled data and the subset of 1400 images that have labels as our labeled data. Once again, the validation data is left untouched.

For speed of training, the Pascal VOC images were resized to 128 by 128. A simple random vertical flipping of the image is applied to the Pascal VOC dataset.

In our experiments, we trained and tested four models.

The first was simply a multilayer perception for classification on MNIST. This will be referred to as the "MNIST Baseline." It was trained using only the 100 labeled binarized images described above. It was trained for 10 epochs with ADAM[10] as an optimizer and binary cross-entropy as a loss. Note, however, that each epoch iterates through the data 590 time so that training will be comparable to the other models. It had two hidden layers, each with 1000 neurons. No batch normalization[11] or dropout[12] was used. It achieved a 75% accuracy on the validation set.

The second was an M2-VAE for classification on MNIST. This will be referred to as the "MNIST M2-VAE." It was trained by iterating through the 59,000 unlabeled images, 100 at a time, training on the 100 labeled data at the same time. This results in a total batch size of 200. It was trained for 10 epochs with ADAM as an optimizer and (7) as a loss. Alpha was chosen to be 10. Each of the MLP's in (8) had 2 hidden layers with 1000 neurons in each. The $MLP's$ for $q_\theta(z|x,y)$ shared weights, except for the output layer. The dimension of z was chosen to be 300. As noted in [3], training is highly variant. Because of this, we trained 10 models and chose the best as our final model. It achieved an 86% accuracy on the validation set.

The third was simply a Deeplabv3+ with a MobileNet backbone for semantic segmentation on Pascal VOC. This will be referred to as the "Pascal VOC Baseline." It was trained by iterating through the 1400 labeled images, 8 at a time. This results in a batch size of 8. It was trained for 10 epochs with ADAM as an optimizer and binary cross-entropy as a loss. The model was modified by removing the batch normalization and dropout. It achieved a 54% IOU coefficient on the validation set.

The fourth was an M2-VAE for semantic segmentation on Pascal VOC. This will be referred to as the "Pascal VOC M2-VAE." It was trained by iterating through the 14000 unlabeled images 8, at a time, while training on 8 of the labeled images at the same time. This results in a batch size of 16. It was trained for 10 epochs with ADAM as an optimizer and (7) as a loss. Alpha was chosen to be 1000. Each of the $DL's$ in (12) was chosen to be a Deeplabv3+ with MobileNet backbone. Once again, the DL's for $q_\phi(z|x,y)$ shared weights, except for the output layer. We trained only one of these models. It achieved a 58% IOU coefficient on the validation set.

5 Further Improvements

The performance increase for the semantic segmentation model over the baseline was relatively small. This is most likely due to many of the simplifying assumptions that were made, most notably, the assumptions made about $p(y)$. One possible way of remedying this is to make a separate discriminatory model that takes a pixel-wise mask as input and outputs the probability of it occurring. It may be possible to introduce parameters for this distribution and train them in the same way that we do theta. We could also train these new parameters using an adversarial training procedure, such as the one described in [13].

Another possible improvement would be to model the pixel map with the more appropriate Categorical distribution for each pixel instead of the Multivariate Bernoulli distribution used now. In the same vein, solving the stability issues with using a Gaussian input, could improve the performance sizably for three channel images. Some more obvious improvements would be to simply train at the original resolution, use more powerful models, and train for more epochs.

As noted in [3] gradually "turning on" the variational components of the lower bound decreases the amount of "dead" z dimensions thus increasing performance. Because we had both labeled and unlabeled data in the same batch, we removed any batch normalization layers. If one could find a way to add them back in, such as by using multiple models with shared parameters, this could increase performance.

What was not noted in (9),(10) and (11) was that taking more than one sample and averaging the results, as described in [2], leads to a better estimate of the gradient. It was noted in [3] that taking ten samples lead to the best performance. Of course, this could be done here.

In the original paper for the M2-VAE, the M1+M2 VAE model was proposed as well as the M2-VAE. This model performed significantly better than the M2-VAE at the expense of losing the ability to train the model end to end. The Skip Deep Generative Model performs better then the M1+M2-VAE, can be trained end to end, and does not suffer from the high variance that the M1+M2-VAE does. This comes at the cost of implementation complexity. Finally, the Importance Weighted Bound[14] could have been employed.

References

[1] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille, "Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[2] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," 2014.

[3] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther, "Auxiliary deep generative models," 2016.

[4] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," 2016.

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," 2018.

[6] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.

[7] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic backpropagation and approximate inference in deep generative models," 2014.

[8] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *ATT Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results." http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[11] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015.

[12] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[14] Y. Burda, R. Grosse, and R. Salakhutdinov, "Importance weighted autoencoders," 2015.