Contrastive losses

Models trained with contrastive losses work as follows.

There is an embedding from data to feature space, which consists of unit vectors,

$f : \mathcal{X} \to F = S^d = \{x \in \mathbb{R}^d \mid \|x\|_2 = 1\}$

The embedding has a natural distance,

$d^2(x, y) = \|x - y\|^2/2 = x^2/2 - x \cdot y + y^2/2 = 1 - x \cdot y = 1 - s(x, y)$

which simplifies, since the vectors are unit norm. The corresponding similarity is $s(x, y) = x \cdot y$, so $d(x, y)^2 = 1 - s(x, y)$.

The embedding comes from using data augmentation. Augmented images should be classified the same (with high probability) as the original images. Different images should have a low similarity.

The data augmentation is designed to erase nuisance features, and keep features which are relevant to downstream applications.

Step 0

- Measure the statistics of $d(x, y)$ for
  - similar images: i.e. average distance (or distance squared or similarity) for the two views of x
  - different images from same class, i.e. average distance (or distance squared) for the two views of x_1 and x_2.
  - different images from different classes.
- Redo the classification (final layer) with
  1. Linear classifier (how they did it:$c_i(x) = w_i * f(x) \quad \min_w KL - SM(c_k(x))$ where k is the correct label.
  2. Linear classifier with $\lambda\|w\|^2$ penalty (so the weight is smaller): $\min_w KL-SM(c_k(x)) + \lambda |w|^2
  3. Define $z(x) = f(x)/\|f(x)\|$. Classifier with $d^2(x, w) = \|z - w\|^2/2$, so for each class, training a w to be close to $x/\|x\|$ (by the algebra above, this should be almost the same as linear classifier). note: now w will automatically be norm 1!
     1. Details on part 3. $g(x, w) = g_1(x, w_1), \ldots g_K(x, w_k)$. Inference: given x, evaluate $g(x, w)$. Classification: $\arg\min_i \|z - w_i\|^2$

Confidence

Classification is done by using a small number of labelled images, then defining a linear (please clarify defnition) classifier, $c(x) = n \cdot f(x)$.
I interpret this to mean that for each class, there is a typical (or mean) example with features $\bar{f}_i$ and then the classifier strength is simply $c_i(x) = \bar{f}_i \cdot f(x)$, based on similarity. Then we have a classifier for each class, and use the strongest one to classify. Confidence should be measured by similiary: $c_i(x) = 1$ would be the highest possible confidence.

Adversarial perturbations

An adversarial perturbation is a perturbation v of small norm so that x+v has a different classification. As shown in previous work, the $\delta = \|v\|_2$ can be estimated by

$\delta \leq \frac{c_i(x) - c_{\max}(x)}{G}$

The ratio of : the gap between the value of the correct classification and the next largest one, and a bound on the gradient of the map, which in this (since the classifier is normalize to have graident 1) should be a bound on $\|\nabla f(x)\|$

Currently, there is nothing in training to make this bound small - except the imbedding pushing things away should already have some good properties.

Regularization

We performed gradient regularization (similar to adversarial training) on these models, which also improved the gradient norm.

Research Plan

1. Measure the gradient norms of the models, on training and test data (we have simple code to do this). Plot a histogram of the gradients. Also plot a histogram (actually a CDF) of the RHS of the bound term above: gap over gradient.
2. Attack the models using standard attack (PGD) and later log-barrier attack (which is slower but maybe strong). Compare the attack curve to the bound
3. Train a model to be robust using
    1. A new form of Data Augmentation, which is to simply add gaussian noise (not smoothing) to the image - this should train the model to be robust.
    2. Tychonoff regualrization (to have small gradients, using Chris' code if possible). This should be a small model, e.g. MNIST, then CIFAR-10.
    3. New ideas: Could also add a smaller amount of noise at each layer of the model - this could work, Data Augmentation at intermediate features...
4. Compare the robustness of the models.

Goal:

```
1. These models are already better than standard CNN out of the box.
```

2. A simple form of Data Augmentation (adding noise) included in training will make them more robust.