Ideas:

1. **Dataset Augmentation:** Other approaches focus on generating images that are similar to existing data (calculate loss between augmented and real image) - applying the ideas from dataset distillation/condensation, we can also generate new synthetic images that are not similar to original images but contain a large amount of information regardless (can apply idea from soft label dataset distillation, where one image contains information about many different classes - the label of real images should be one hot encoding). For instance, instead of generating a new image for a specific class using 2 existing images from that class, can we generate a new image + its soft label from existing images taken from several classes? If these new generated images are used in conjunction with existing real images, is the performance improved and is overfitting reduced?
2. **Dataset Condensation for class imbalance:** do experiments to verify the effect of DC and other dataset distillation methods on imbalanced data → do the less represented classes get misclassified more using condensed data and to what extent, is the relationship linear etc.? Verify whether using dataset condensation with soft labels can alleviate this problem to a certain extent / Verify whether using augmentation methods can alleviate this problem
3. **Dataset Distillation:** Is it possible to "summarise" the large dataset using matrix factorization? Each column is approximated by the linear combination of the dictionary matrix and corresponding column of the code matrix. ??????

Data augmentation (non-traditional)

Traditional = cropping, flipping, colour etc.

A survey on Image Data Augmentation for Deep Learning

Augmentation using deep learning:

Feature space augmentation
The lower-dimensional representations of image data in fully-connected layers can be extracted and isolated. Konno and Iwazume find a performance boost on CIFAR-100 from 66 to 73% accuracy by manipulating the modularity of neural networks to isolate and refne individual layers after training. Lower-dimensional representations found in high-level layers of a CNN are known as the feature space. DeVries and Taylor presented an interesting paper discussing

augmentation in this feature space. This opens up opportunities for many vector operations for Data Augmentation.

Alleviate problems with class imbalance
SMOTE is a popular augmentation used to alleviate problems with class imbalance. This technique is applied to the feature space by joining the k nearest neighbors to form new instances.

The Effectiveness of Data Augmentation in Image Classification using Deep Learning (2017)

Uses 2 neural networks: 1 that generates the augmented image and the other that does the classifying

1. At training time, this neural net takes in two random images from the training set and outputs a single ”image” so that this image matches either in style or in context with a given image from the training set.
2. This output, which represents an augmented image produced by the network, is fed into the second classifying network along with the original training data. The training loss is then back-propagated to train the augmenting layers of the network as well as the classification layers of the network。
3. In test time, images from the validation or test set are run through only the classification network. The motivation is to train a model to identify the best augmentations for a given dataset.
4. The classification loss at the end of the network is a cross entropy loss on the sigmoids of the scores of classes. An additional loss is computed at the end of the augmentation network to regulate how similar the augmented image should be to the input image. The overall loss is a weighted sum of these two losses.
    a. Content loss
    b. Style loss via gram matrix
    c. No loss at this layer

Takes almost 3x the amount of time than traditional augmentation techniques (cropping etc)

The Neural Augmentation approach takes in two random images from the same class. The prepended augmentation net maps them into a new image through a CNN with 5 layers, each with 16 channels, 3×3 filters, and ReLU activation functions. The image output from the augmentation is then transformed with another random image via Neural Style Transfer. This style transfer is carried out via the CycleGAN [92] extension of the GAN [31] framework. These images are then fed into a classification model and the error from the classification model is back propagated to update the Neural Augmentation net. Te Neural Augmentation network uses this error to learn the optimal weighting for content and style images between different images as well as the mapping between images in the CNN

Smart Augmentation Learning an Optimal Data Augmentation Strategy

Unsupervised vs supervised augmentation

- Unsupervised: data augmentation is done regardless of the label of the sample (adding noise, rotating, flipping etc)
- Supervised: mix different samples with the same label in feature space in order to generate a new sample with the same label. The generated sample has to be recognizable as a valid data sample, and also as a sample representative of that sample class. Labels of data are used to generate new samples.

Use network A to learn the best data augmentation to train network B

1. Network A accepts several inputs from the same class (either random, or clustering in the pixel or feature space), generate an output which approximates data from that class
    a. Minimise the loss function between generated image and another real image from the same class
2. Output of network A is used to train network B
3. Error back propagates from network B to network A → tunes network A to generate the best augmentations for network B
4. Total loss = linear combination of loss between generated image by A and image from same class and loss of output vs real labels for network B

This is done by having two networks, Network-A and Network-B. Network-A is an augmentation network that takes in two or more input images and maps them into a new image or images to train Network-B. The change in the error rate in Network-B is then back propagated to update Network-A. Additionally another loss function is incorporated into Network-A to ensure that its outputs are similar to others within the class. Network-A uses a series of convolutional layers to produce the augmented image. The conceptual framework of Network-A can be expanded to use several Networks trained in parallel. Multiple Network-As could be very useful for learning class-specific augmentations via meta-learning

Learning from Simulated and Unsupervised Images through Adversarial Training

- Learning from synthetic images can be problematic due to a gap between synthetic and real image distributions - synthetic data is often not realistic enough, leading the network to learn details only present in synthetic images and failing to generalize well on real images - lack of realism causes models to overfit to unrealistic details in synthetic images
- Many efforts have explored using synthetic data for various prediction tasks, including gaze estimation [43], text detection and classification in RGB images [9, 15], font recognition [42], object detection [10, 27], hand pose estimation in depth images [38,

37], scene recognition in RGB-D [11], semantic segmentation of urban scenes [31], and human pose estimation [26, 3, 18, 14, 28, 30].

Goal: improve realism of synthetic images from simulator using unlabeled real data

1. A synthetic image is generated with a black box simulator and is refined using the refiner network. To add realism, we train our refiner network using an adversarial loss, similar to Generative Adversarial Networks (GANs) [8], such that the refined images are indistinguishable from real ones using a discriminative network.
2. To preserve the annotations of synthetic images, we complement the adversarial loss with a self-regularization loss that penalizes large changes between the synthetic and refined images.

AutoAugment: Learning Augmentation Strategies from Data

- Aim to automate the process of finding an effective data augmentation policy for a target dataset.
- Each policy expresses several choices and orders of possible augmentation operations, where each operation is an image processing function (e.g., translation, rotation, or color normalization), the probabilities of applying the function, and the magnitudes with which they are applied.
- Use a search algorithm to find the best choices and orders of these operations such that training a neural network yields the best validation accuracy.
- Use Reinforcement Learning as the search algorithm

AutoAugment is a Reinforcement Learning algorithm that searches for an optimal augmentation policy amongst a constrained set of geometric transformations with miscellaneous levels of distortions. The AutoAugment approach learns a policy which consists of many sub-policies, each sub-policy consisting of an image transformation and a magnitude of transformation.

Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition

Natural data blending – each of the image-layers are blended with a randomly-sampled crop of an image from the training datasets of ICDAR 2003 and SVT. The amount of blend and alpha blend mode (e.g. normal, add, multiply, burn, max, etc.) is dictated by a random process, and this creates an eclectic range of textures and compositions. The three image-layers are also blended together in a random manner, to give a single output image

DATASET AUGMENTATION IN FEATURE SPACE

Imbalanced Data

To do:

1. Metrics for evaluating/comparing performance on imbalance datasets
2. Distribution of classes that are imbalanced (long tailed, normal etc, which ones are most useful to investigate)
    a. Data distribution factors: class overlap, imbalance ratio, size of training instances, noisy data, small disjunctions
3. Papers explaining math behind why methods fail on imbalance datasets
4. Effect on condensed images and whether you can explain it
5. Incorporate weights into the data condensation algorithm to balance out the imbalance (read up on papers that involve the cost-sensitive approaches) OR the threshold thing → particle swarm optimisation (read up more to understand)

http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=047D2F2C49B123F3AC37A700726A5D1A?doi=10.1.1.711.8214&rep=rep1&type=pdf

What constitutes unbalanced data:

1. Concept complexity
2. Training set size
3. Degree of imbalance

Learning from imbalanced data: open challenges and future directions

1. **Data-level methods** - modify the collection of examples/training set to balance distributions and/or remove difficult samples
    a. Oversampling - generate new objects for minority groups
    b. Undersampling - remove examples from majority groups
    c. Random selection of target samples
    d. Maintain structures of groups/generate new data according to underlying distributions
    e. Clean overlapping objects and remove noisy examples
2. **Algorithm-level methods** - directly modify existing learning algorithms to alleviate the bias towards majority objects and adapt them to mining data with skewed distributions

    a. Cost-sensitive approaches - learner is modified to incorporate varying penalty for each of considered groups of examples - assign a higher cost to less represented set of objects to boost its importance during the learning process
    b. One-class learner - focuses on creating a data description for target group, eliminates bias towards any group
3. **Hybrid methods** - combine the advantages of two previous groups
    a. Sampling + cost-sensitive learning

Binary problems

Slight imbalance

1. Analyse the neighbourhood of each minority class example and assign it to one of the four predefined groups: safe, borderline, rare and outliers → tackle difficult examples
    a. Currently based on k-nearest neighbours, but this implies uniform distribution of data.
    b. Future: Adaptive methods that adjusts the size of analysed neighbourhood according to local densities or chunk sizes
2. Vary level of oversampling according to example types, supervise the undersampling in order not to discard important representatives

Extreme imbalance

1. Methods that can predict or reconstruct a potential class structure for minority class
2. Decomposition of original problem into a set of subproblems, each with reduced imbalance ratio
    a. Requires algorithms for meaningful problem division
    b. Algorithms for reconstruction of original extremely imbalanced task

Methods

1. **Classifier's output adjustment**: Recent studies show that weighting or putting a threshold on continuous output of a classifier (known also as support functions or class probability estimates) can often lead to better results than data resampling and may be applied to any conventional classifier
2. **Ensemble learning:** hybridization of bagging, boosting and random forests with sampling or cost-sensitive methods

Multi-label

A single example can be characterized by more than one class label

- SMOTE-based oversampling [8]
- Need for skew-insensitive classifiers that do not require resampling strategies
- Hierarchical multi-label classification tree

- Classifier chains

SMOTE: Synthetic Minority Over-sampling Technique

- Minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbours
- To generate synthetic samples: take the difference between the feature vector and its nearest neighbour, multiply this difference by a random number between 0 and 1, add it to the feature vector → This causes the selection of a random point along the line segment between two specific features
- Synthetic samples causes classifier to create larger and less specific decision regions

ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning

- The essential idea of ADASYN is to use a weighted distribution for different minority class examples according to their level of difficulty in learning, where more synthetic data is generated for minority class examples that are harder to learn compared to those minority examples that are easier to learn.

On Multi-Class Cost-Sensitive Learning

Rescaling is a general approach which can be used to make any cost-blind learning algorithms cost-sensitive. The principle is to enable the influences of the higher-cost classes be bigger than that of the lower-cost classes.

ODOC-ELM: Optimal decision outputs compensation-based extreme learning machine for classifying imbalanced data

- In recent years, a number of methods have been developed to deal with class imbalance problem, including resampling [18–23], cost-sensitive learning [24–26], decision boundary moving [27–29], ensemble learning [30–35], active learning [36] and one class classifier [37,38].

- Imbalanced data can be found almost everywhere, from biomedical applications [16] to network intrusion detection [17].
- Zong et al. [15] profited from the idea of cost-sensitive learning to present a weighted ELM classifier (WELM). By designating different penalty factors for the training errors belonging to different categories, the performance of the minority classes could be highlighted.
- For multiclass data, there are multiple minority classes, thus it is a multivariate optimization problem. Particle swarm optimization (PSO) algorithm [47,48] is used to find the optimal combination of compensations. In addition, readers are also encouraged to replace them with other optimization algorithms in their practical applications.
- 30 binary-class imbalanced data sets and 12 multiclass imbalanced data sets randomly acquired from the Keel data repository [49].

Classifiers that pursue the minimization of training errors are apt to be destroyed by imbalanced class distributions, including Naïve Bayes classifiers [52], K nearest neighbors (KNN) classifiers [53], multilayer perceptrons (MLP) [54] and support vector machines (SVM) [29]. The fact is that the training errors often appear in the overlapping region between two different categories, and the desired classification boundary is the center of the class overlapping region. In this region, however, the number of instances belonging to the majority class is much more than that of the minority class. To guarantee the minimization of training errors, the minority class has to sacrifice more than the majority class

not all imbalanced classification tasks are harmful, and for those unharmful imbalanced classification tasks, adopting specific class imbalance learning methods might increase unnecessary temporal and/or spatial costs. In fact, the damage is related to multiple potential data distribution factors, including class overlap, imbalance ratio, the size of training instances, noisy data and small disjunctions [52,55–57].

Besides RUS, ROS and SMOTE, other resampling strategies include random walk oversampling (RWOS) [20], MLSMOTE [21], clustering-based undersampling [58]

Zhou and Liu [27] proposed a method named threshold moving that first adopts 0–1 outputs to train MLP, then normalizes the sum of outputs as 1, and finally multiplies by the corresponding cost for the output of each class. They found that the threshold moving is resource-saving and relatively efficient in class imbalance tasks
. Actually, the purpose of decision output compensation is to push the classification boundary back towards the majority class.

1. Resampling → oversampling, undersampling, synthetic
2. Weighted algorithm → assign costs
3. Moving decision boundary of classifier

F-measure and G-mean can better represent the desired classification behavior than overall accuracy. F-measure and G-mean are functions of the confusion matrix

$$\mathrm{F-measure} = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

$$\mathrm{G-mean} = \sqrt{\mathrm{TPR} \times \mathrm{TNR}}$$

where Precision, Recall, TPR and TNR are further defined as:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

$$\mathrm{Recall} = \mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

For a multi-class imbalance problem, the problem becomes more complex [59,60]. Except the class with the most instances, the outputs of all other classes need to be compensated by different thresholds. Therefore, it becomes a multivariate optimization problem. To solve this optimization problem, particle swarm optimization (PSO) algorithm [47,48] is adopted. PSO is a population-based stochastic optimization technique, inspired by the social behavior of bird flocking. Specifically, PSO is originally designed to address continuous optimization problem, thus it can be directly used to deal with our problem without any modifications. During the optimization process of PSO, each particle dynamically changes its position and velocity by recalling its historical optimal position (pbest) and observing the position of the optimal particle (gbest). On each round, the position of each particle is updated by:

Te imbalanced dataset problems become more complicated in multi-class imbalanced classifcation tasks, in which there may be multiple minority and majority classes that cause skew data distribution. In this case, for example, a class may be a minority one when compared to some other classes, but a majority of the rest of them [13]. Terefore, many new challenges are imposed which do not exist in two-class cases. Fortunately, some strategies such as decomposition methods and introducing new loss functions have been proposed to deal with multi-class classifcation problem [14, 15].

Multi-class imbalanced big data classification on Spark