Precancerous Lesion Detection using AI/ML

Aaryan Dangi[1], Nikhil Verma[2], Rahul Deepak[3], Keshava Kotagiri[4], Rajkumar[5]
[1]*210050001*, [2]*210050109*, [3]*210050090*, [4]*210050088*, [5]*210050129*

***Abstract*—In this project, we're addressing the challenge of creating an efficient skin cancer diagnosis tool using image classification without relying on expensive labeled datasets. Such a tool could enable reliability in early cancer detection, potentially increasing survival rates by 90% for skin cancers. To overcome the hurdles of limited labeled data and imbalanced datasets, we're exploring various approaches. We start with a supervised model as a baseline, then implement the well-established MixMatch algorithm for semi-supervised learning. Additionally, we customize MixMatch with focal loss and adjustable weights to better handle long-tailed datasets. Our goal is to identify key features influencing classification accuracy and analyze the impact of methods like consistency regularization, entropy minimization, and variable weights in long-tailed semi-supervised setups. The findings can contribute to improving real-world model suitability for cancer detection and related fields.**

***Index Terms*—MixMatch, SSL**

I. Introduction

Skin lesions represent abnormal changes in the structure or appearance of the skin, exhibiting a wide range of causes and characteristics. Although many lesions are benign, some may indicate skin cancer or other serious conditions. Dermatologists play a crucial role in evaluating these lesions to ascertain whether they are harmless or require further investigation.

Timely detection significantly improves the likelihood of successful treatment and reduces the risk of complications. Machine learning (ML) algorithms offer an effective means to analyze dermatoscopic images automatically, aiding in the early identification of potential malignancies.

However, the training of neural networks for automated diagnosis of pigmented skin lesions faces challenges due to the limited size and lack of diversity in available dermatoscopic image datasets. Existing datasets often lack representation across all significant diagnostic categories, including Actinic keratoses and intraepithelial carcinoma (akiec), basal cell carcinoma (bcc), benign keratosis-like lesions (solar lentigines / seborrheic keratoses and lichen-planus like keratoses, bkl), dermatofibroma (df), melanoma (mel), melanocytic nevi (nv), and vascular lesions (angiomas, angiokeratomas, pyogenic granulomas, and hemorrhage, vasc).

Dermatoscopy, a widely employed diagnostic technique, enhances the diagnosis of pigmented skin lesions compared to unaided eye examination. Dermatoscopic images serve as a suitable source for training artificial neural networks for automated diagnosis.

The challenge lies in the need for a large number of annotated images for training neural-network-based diagnostic algorithms. However, the availability of high-quality dermatoscopic images with reliable diagnoses is limited, often restricted to only a few disease classes. Previous studies faced limitations in sample size and a lack of dermatoscopic images beyond melanoma or nevi. Advances in graphics card capabilities and machine learning techniques are setting new benchmarks, fostering expectations that automated diagnostic systems will soon be capable of diagnosing various pigmented skin lesions without the need for human expertise.

II. Semi Supervised Learning

Semi-supervised learning is a machine learning paradigm that falls between supervised learning (where the model is trained on labeled data) and unsupervised learning (where the model deals with unlabeled data). In semi-supervised learning, the model is trained on a dataset that contains a mix of labeled and unlabeled examples. The goal is to leverage the limited labeled data to improve the model's performance on the larger set of unlabeled data.

In many real-world scenarios, obtaining labeled data can be expensive, time-consuming, or impractical. This is true especially in the medical world, where obtaining labeled data is often challenging due to privacy concerns and the high cost of manual annotation. Semi-supervised learning allows you to make the most of the limited labeled data by combining it with a larger set of unlabeled data. Our project leverages semi-supervised learning to address this limitation. By combining the limited labeled data with a larger pool of unlabeled medical data, we aim to enhance model performance, reduce the need for extensive manual labeling, and adapt to the evolving nature of medical datasets.

III. Model

Our project employs a pre-trained ResNet-50 model as the foundation for lesion classification. ResNet-50 employs a deep Convolutional Neural Network (CNN) architecture known for its success in image classification tasks. Its distinctive feature is the use of residual

blocks, allowing the training of very deep networks while mitigating the vanishing gradient problem. Moreover, leveraging a pre-trained model accelerates convergence and enhances the model's ability to capture intricate features in medical images.

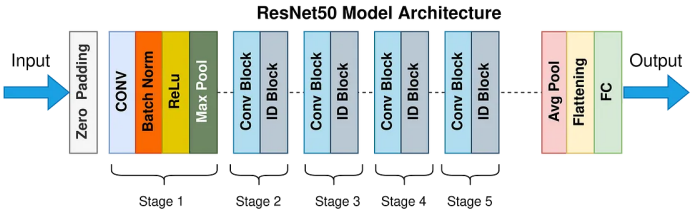

Fig. 1. Resnet 50 Architecture

To tailor ResNet-50 to our specific task, we replaced its final fully connected layer with a custom architecture. The new configuration comprises a linear layer with 256 units, followed by a rectified linear unit (ReLU) activation function, a dropout layer for regularization, and a final linear layer with the number of classes in our dataset.

IV. Building upto MixMatch

We begin by presenting established semi-supervised learning (SSL) methods, with a primary focus on the current state-of-the-art approaches that serve as the foundation for MixMatch's development.

In the subsequent discussion, we will use the term $p_{\text{model}}(y|x;\theta)$ to represent a generic model. This model generates a distribution over class labels y based on an input x and is characterized by parameters θ.

The fundamental aspects are:

A. Consistency Regularization

A widely employed regularization technique in supervised learning is data augmentation, involving input transformations assumed to preserve class semantics. For instance, in image classification, common practices include elastically deforming or adding noise to input images, altering pixel content without changing the label. Essentially, this artificially expands the training set by generating a virtually infinite stream of new, modified data.

Consistency regularization extends data augmentation to semi-supervised learning, leveraging the concept that a classifier should produce the same class distribution for an unlabeled example even after augmentation. More precisely, consistency regularization ensures that an unlabeled example x should be classified the same as its augmentation, $Augment(x)$.

In its simplest form, for unlabeled points x, prior work introduces the loss term:

$$\|p_{\text{model}}(y|\text{Augment}(x);\theta) - p_{\text{model}}(y|\text{Augment}(x);\theta)\|_2^2$$

It's important to note that $\text{Augment}(x)$ is a stochastic transformation, making the two terms in equation non-identical. MixMatch employs a version of consistency regularization by utilizing standard data augmentation techniques for images, such as random horizontal flips and crops.

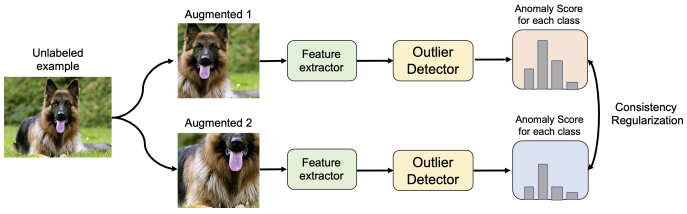

Fig. 2. Consistency Regularization

B. Entropy Minimization

Many semi-supervised learning methods share a fundamental assumption: the classifier's decision boundary should avoid traversing high-density regions within the marginal data distribution. The rationale behind this assumption lies in the belief that high-density regions are more likely to harbor informative and reliable data points.

The prospect of decision boundaries passing through high-density regions introduces potential ambiguity, as these regions often encompass a mixture of different class instances. To mitigate this, one approach is to ensure that the classifier produces low-entropy predictions on unlabeled data. Low-entropy predictions indicate high confidence in the model's classifications. By enforcing low entropy, the model is encouraged to make more decisive predictions on unlabeled data, reducing ambiguity and providing clearer indications of class membership.

This is explicitly achieved by incorporating a loss term that minimizes the entropy of $p_{\text{model}}(y|x;\theta)$ for unlabeled data x.

MixMatch takes a different route, implicitly achieving entropy minimization. This is accomplished through the application of a "sharpening" function on the target distribution for unlabeled data.

C. Traditional Regularization

Regularization involves imposing constraints on a model to prevent it from memorizing the training data excessively, with the goal of enhancing its generalization

to unseen data. Weight decay is utilized as a regularization technique, penalizing the L2 norm of the model parameters.

Additionally, in MixMatch, we incorporate MixUp to promote convex behavior "between" examples. MixUp serves dual roles as both a regularizer (applied to labeled datapoints) and a semi-supervised learning method (applied to unlabeled datapoints).

V. MixMatch

Many recent approaches for semi-supervised learning add a loss term which is computed on unlabeled data and encourages the model to generalize better to unseen data. In much recent work, this loss term falls into one of three classes: entropy minimization -which encourages the model to output confident predictions on unlabeled data; consistency regularization—which encourages the model to produce the same output distribution when its inputs are perturbed; and generic regularization—which encourages the model to generalize well and avoid overfitting the training data.

MixMatch is an algorithm designed for semi-supervised learning, which leverages both labeled and unlabeled data during the training process. MixMatch introduces a unified loss term for unlabeled data that seamlessly reduces entropy while maintaining consistency and remaining compatible with traditional regularization techniques. In MixMatch, the primary idea is to generate pseudo-labeled data for unlabeled examples by combining the model's predictions on these examples with augmentations of other examples. The algorithm aims to minimize a unified loss function that encourages consistency between the model's predictions on the original and augmented examples. This approach helps the model generalize better to unseen data, especially when labeled data is limited.

VI. MixMatch Algorithm

For a batch X containing labeled examples with one-hot targets (representing one of L possible labels), along with an equally-sized batch U comprising unlabeled examples, MixMatch generates a modified batch of augmented labeled examples X' and a batch of augmented unlabeled examples with "guessed" labels U'. These augmented batches, U' and X', are subsequently employed in the computation of distinct labeled and unlabeled loss terms. Next, we describe each part of MixMatch.

A. Augmentation

Augmentation refers to the technique of applying various transformations or modifications to the original data to create new, slightly altered instances. The primary goal of data augmentation is to increase the diversity of the training dataset, helping the model generalize better to unseen data and improving its robustness.

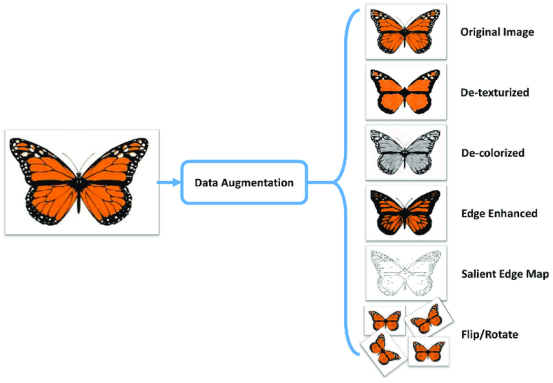

Fig. 3. Augmentation

As is typical in many SSL methods, we use data augmentation both on labeled and unlabeled data. For each x_b in the batch of labeled data X, we generate a transformed version $\hat{x}_b = \text{Augment}(x_b)$ (Algorithm 1, line 3). For each u_b in the batch of unlabeled data U, we generate K augmentations $\hat{u}_{b,k} = \text{Augment}(u_b)$, $k \in (1, \ldots, K)$ (Algorithm 1, line 5). We use these individual augmentations to generate a "guessed label" q_b for each u_b.

B. Label Guessing

For each unlabeled example in U, MixMatch produces a "guess" for the example's label using the model's predictions. The final "guessed label" is often derived from the combination or aggregation of these predictions. This guess is later used in the unsupervised loss term. To do so, we compute the average of the model's predicted class distributions across all the K augmentations of u_b by

$$\overline{q_b} = \frac{1}{K} \sum_{k=1}^{K} p_{\text{model}}(y|\hat{u}_{b,k}; \theta)$$

in Algorithm 1, line 7. Using data augmentation to obtain an artificial target for an unlabeled example is common in consistency regularization methods.

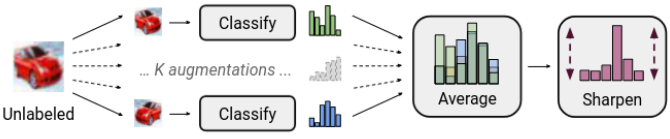

Fig. 4. Label Guessing and Sharpening

Algorithm 1 MixMatch takes a batch of labeled data $\mathcal{X}$ and a batch of unlabeled data $\mathcal{U}$ and produces a collection $\mathcal{X}'$ (resp. $\mathcal{U}'$) of processed labeled examples (resp. unlabeled with guessed labels).

Input: Batch of labeled examples and their one-hot labels $\mathcal{X} = \big((x_b, p_b); b \in (1, \dots, B)\big)$, batch of unlabeled examples $\mathcal{U} = \big(u_b; b \in (1, \dots, B)\big)$, sharpening temperature T, number of augmentations K, Beta distribution parameter α for MixUp.
for $b = 1$ **to** B **do**
  $\hat{x}_b = \text{Augment}(x_b)$ *// Apply data augmentation to x_b*
  for $k = 1$ **to** K **do**
    $\hat{u}_{b,k} = \text{Augment}(u_b)$ *// Apply k^{th} round of data augmentation to u_b*
  end for
  $\bar{q}_b = \frac{1}{K} \sum_k \text{p}_{\text{model}}(y \mid \hat{u}_{b,k}; \theta)$ *// Compute average predictions across all augmentations of u_b*
  $q_b = \text{Sharpen}(\bar{q}_b, T)$ *// Apply temperature sharpening to the average prediction (see eq. (7))*
end for
$\hat{\mathcal{X}} = \big((\hat{x}_b, p_b); b \in (1, \dots, B)\big)$ *// Augmented labeled examples and their labels*
$\hat{\mathcal{U}} = \big((\hat{u}_{b,k}, q_b); b \in (1, \dots, B), k \in (1, \dots, K)\big)$ *// Augmented unlabeled examples, guessed labels*
$\mathcal{W} = \text{Shuffle}\big(\text{Concat}(\hat{\mathcal{X}}, \hat{\mathcal{U}})\big)$ *// Combine and shuffle labeled and unlabeled data*
$\mathcal{X}' = \big(\text{MixUp}(\hat{\mathcal{X}}_i, \mathcal{W}_i); i \in (1, \dots, |\hat{\mathcal{X}}|)\big)$ *// Apply MixUp to labeled data and entries from $\mathcal{W}$*
$\mathcal{U}' = \big(\text{MixUp}(\hat{\mathcal{U}}_i, \mathcal{W}_{i+|\hat{\mathcal{X}}|}); i \in (1, \dots, |\hat{\mathcal{U}}|)\big)$ *// Apply MixUp to unlabeled data and the rest of $\mathcal{W}$*
return $\mathcal{X}', \mathcal{U}'$

Fig. 5. Mixmatch Algorithm

C. Sharpening

Sharpening involves adjusting the predicted class probabilities for unlabeled examples to increase confidence. This adjustment is accomplished by applying a 'sharpening' function to the predicted probabilities, aiming to accentuate the most probable classes and downplay less likely ones, ultimately refining the model's predictions to be more focused and certain.

In the label guessing process, we enhance confidence inspired by the effectiveness of entropy minimization in semi-supervised learning (SSL). To decrease the entropy of the label distribution, we utilize a sharpening function applied to the average prediction over augmentations, denoted as $\bar{q}_b$. In practice, we employ the standard method of adjusting the "temperature" of this categorical distribution, described by the operation:

$$\text{Sharpen}(p, T)_i := \frac{p_i^{1/T}}{\sum_{j=1}^{L} p_j^{1/T}} \tag{1}$$

Here, p represents some input categorical distribution, where in MixMatch, it is the average class prediction over augmentations ($\bar{q}_b$), and T (temperature) serves as a hyperparameter. Lowering the value of T encourages the model to produce lower-entropy predictions.

D. MixUp

MixUp is a data augmentation technique used for both labeled and unlabeled examples in the MixMatch semi-supervised learning framework. The MixUp process involves combining pairs of examples and their corresponding labels to create augmented samples. This is done by linearly interpolating between two examples and their labels, effectively blending them together.

We incorporate MixUp as a semi-supervised learning technique, departing from previous approaches in SSL by extending its application to both labeled and unlabeled examples, including those with label guesses generated. To ensure compatibility with our distinct loss terms, we use a slightly modified version of MixUp.

For a pair of examples with their corresponding label probabilities, denoted as (x_1, p_1) and (x_2, p_2), we compute (x', p') using the following equations:

$$\begin{aligned} \lambda &\sim \text{Beta}(\alpha, \alpha) \\ \lambda' &= \max(\lambda, 1 - \lambda) \\ x' &= \lambda' x_1 + (1 - \lambda') x_2 \\ p' &= \lambda' p_1 + (1 - \lambda') p_2 \end{aligned}$$

Here, α is a hyperparameter. It's important to note that it introduces $\lambda' = \max(\lambda, 1 - \lambda)$, which differs from vanilla MixUp that omits this step (setting $\lambda' = \lambda$).

The implementation of MixUp follows this procedure:

1) We first collect all augmented labeled examples with their labels and all unlabeled examples with their guessed labels into
   - $\hat{X} = \{(\hat{x}_b, p_b) \mid b \in (1, \dots, B)\}$)

- $\hat{U} = \{(\hat{u}_{b,k}, q_b) \mid b \in (1, \ldots, B), k \in (1, \ldots, K)\}$

2) We combine these collections and shuffle the result to form W which serves as a data source for MixUp.
3) For each (x_i, p_i) pair in $\hat{X}$, we compute MixUp (x_i, W_i) and add the result to the collection $X^{'}$
4) We compute $U_i' = \text{MixUp}(\hat{u}_i, W_{i+|\hat{X}|})$ for $i \in (1, \ldots, |\hat{U}|)$, intentionally using the remainder of W that was not used in the construction of $X^{'}$

To summarize, MixMatch transforms X into $X^{'}$, a collection of labeled examples which have had data augmentation and MixUp (potentially mixed with an unlabeled example) applied. Similarly, U is transformed into $U^{'}$, a collection of multiple augmentations of each unlabeled example with corresponding label guesses.

VII. Loss Function

The processed collections U' and X' obtained from the MixMatch algorithm are subsequently employed to calculate distinct loss terms for labeled and unlabeled data. More precisely, the collective loss L for semi-supervised learning is formally defined as:

$$U', X' = \text{MixMatch}(U, X, T, K, \alpha)$$

$$L_X = \frac{1}{|X'|} \sum_{x,p \in X'} H(p, p_{\text{model}}(y|x;\theta))$$

$$L_U = \frac{1}{L|U'|} \sum_{u,q \in U'} \|q - p_{\text{model}}(y|u;\theta)\|_2^2$$

$$L = L_X + \lambda_U L_U$$

Here, $H(p, q)$ represents the cross-entropy between distributions p and q, and the hyperparameters T, K, α, and λ_U.

The combined loss function L integrates conventional cross-entropy loss (H), calculated between true labels and model predictions from X', with squared L_2 loss computed on predictions and guessed labels from U'. The choice of employing L_2 loss stems from its bounded nature and reduced sensitivity to inaccuracies compared to cross-entropy, making it a common choice for unlabeled data loss in semi-supervised learning and a metric for evaluating predictive uncertainty. Importantly, gradients are not propagated through the computation of guessed labels, aligning with standard practices. The MixMatch algorithm utilizes this loss function to improve predictions by minimizing it.

VIII. Training

A. Dataset

In this project, we utilized the HAM-10000 dataset for the classification of lesion images into different skin cancer classes.

The training image dataset was randomly divided into a 20-80 split of labeled and unlabeled images. Additionally, the test dataset, along with its corresponding truth values, was accessible on the website, enabling the reporting of loss and accuracy.

The distribution of training images is imbalanced, with a majority belonging to the ”nv” class. This imbalance suggests that a significant portion of the conducted tests is benign, which is to be expected. However, this makes training our model to predict lesion class significantly more difficult, as it is likely to be rewarded for blind-guessing ”nv” initially, and possibly converge to a local minima.

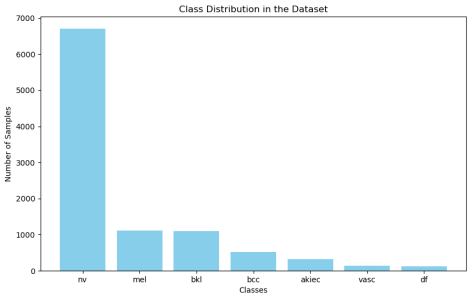

Fig. 6. Class Distribution

B. Implementation

We applied augmentations, including cropping, rotation, flipping, and normalization, to enhance image generalization. Subsequently, a pre-trained ResNet-50 model was trained on labeled images for benchmarking using traditional supervised learning. This benchmark served for comparison with our semi-supervised learning, leveraging a substantial amount of unlabeled images.

For supervised learning, we utilized negative log-likelihood loss, updating the ResNet-50 model’s fully connected layer while keeping other layers frozen.

Predicted images were obtained by softmax of the predicted probability values for each class. In contrast, the MixMatch algorithm, incorporating MixUp and sharpening, was implemented. A weighted cross-entropy loss, addressing dataset imbalance, assigned weights inversely proportional to class representation, preventing bias towards classes with fewer images.

This approach was applied to a fresh ResNet-50 model, modifying the fully connected layer accordingly.

Accuracy was measured on the test dataset for comparative analysis with the supervised learning approach.

C. Hyper parameters

MixMatch employs multiple techniques to predict labels for unlabeled data, introducing various hyperparameters for tuning. The key hyperparameters controlling prediction values include the sharping temperature (T), the number of unlabeled augmentations (K), α in the Beta function for MixUp, and the unsupervised loss weight (λ_μ). Due to the numerous hyperparameters, cross-validation becomes challenging with small validation sets. However, MixMatch's authors suggest fixing most hyperparameters, setting T = 0.5 and K = 2, and only varying α and λ_μ. After tuning, the optimal α value is found to be 0.9.

IX. Results

Supervised models plateau around 65% accuracy, even with increased epochs, aligning with the imbalance in the "nv" class images in our dataset. This imbalance is a concern addressed in the MixMatch section.

For "nv" class images, the supervised model yields high probabilities, but struggles with images from other classes.

MixMatch achieves 62.60% accuracy, indicating room for improvement in image prediction. Notably, when the actual label isn't "nv," the model, while still predicting "nv," shows significantly higher probabilities for the image truly belonging to that class. This suggests potential accuracy enhancement through further tuning, improved augmentations, and a larger dataset.

X. Conclusions

A. Error Analysis

The low accuracy of the model can be attributed to the low number of epochs we are running are model on, lack of hyperparameter tuning which we basically did none of and almost stuck with the original values which we had assumed at the start.

And above all, our model still majorly predicts the majority class i.e. "nv", even after applying the weighted loss and trying oversampling. The accuracy remains nearly unchanged for all the changes tried with this model architecture.

B. Suggested Improvements

The limited compute power over a large dataset really hindered our testing and tuning ability. We suspect that a larger number of epochs, with better hyperparameter tuning, would improve results substantially. Switching pre-trained models from ResNet-50 to a shallower net may help us tune and debug better as well.

After much observation, we were also led to believe that a different architecture which involves supervised training for 10-15 epochs before enabling MixUp and augmentation would likely be more favourable for our model. This change could help prevent convergence to a local minima, as is happening otherwise with our version of modified MixMatch.

Finally, dynamically changing the ratio of labelled and unlabelled images could improve the accuracy as well. Again, it is perhaps best to stick to supervised principles in earlier epochs before incorporating consistency regularization, etc for unlabelled images.

C. Failed attempts

Undersampling gave unsatisfactory results. Oversampling did not have much effect on the result either. Our results were further limited by a lack of effective hyperparameter tuning, due to the excessively large running times. Even the weighted entropy loss , didn't improve the result by much.

References

Tschandl, P., Rosendahl, C. Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. Sci Data 5, 180161 (2018). https://doi.org/10.1038/sdata.2018.161

Tschandl, Philipp, 2018, "The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions", https://doi.org/10.7910/DVN/DBW86T, Harvard Dataverse, V4, UNF:6:KCZFcBLiFE5ObWcTc2ZBOA== [fileUNF]

ISIC Challenge. (n.d.). https://challenge.isic-archive.com/landing/2018/47/

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C. (2019). MixMatch: A Holistic Approach to Semi-Supervised Learning. NeurIPS.

Saito, K., Kim, D., Saenko, K. (2021). OpenMatch: Open-set Consistency Regularization for Semi-supervised Learning with Outliers. arXiv [Cs.CV]. Retrieved from http://arxiv.org/abs/2105.14148

Mukherjee, S. (2022, August 18). The annotated resnet-50. Medium. https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758

Shi, Z., Tonolini, F., Aletras, N., Yilmaz, E., Kazai, G., Jiao, Y. (2023). Rethinking Semi-supervised Learning with Language Models. arXiv [Cs.CL]. Retrieved from http://arxiv.org/abs/2305.13002