Group-6 Report on

TUT Copper Analysis Competition

Group Members:

Ebrahim Afyounian 242702

Aki Ropo: 218620

Sampsa Ikola: 224924

Oskari Jessen-Juhler: 225233

Marko M Leppanen: 240961

Pekka Lempiäinen: 233588

Introduction

This document reports on the material and methods used to participate in the *TUT Copper Analysis Competition* (held on 21 Jan 2016 – Sun 6 Mar 2016) where participants were expected to predict the number of Equal channel angular pressing (ECAP) passes performed on copper samples [1]. Following sections, explain in detail steps taken to perform the predictions. The code implemented in Python is available at: https://github.com/eafyounian/Kaggle_TUT_Copper_Analysis.

Data

Data were composed of color microscope images of copper with different ECAP passes (see fig 1) [2]. Each copper image was 512×512 pixels. Data were organized into *train* and *test* datasets where each contained 540 and 360 images respectively. Training data contained 90 images for each of the following ECAP passes: 0, 1, 3, 4, 6, and 8.

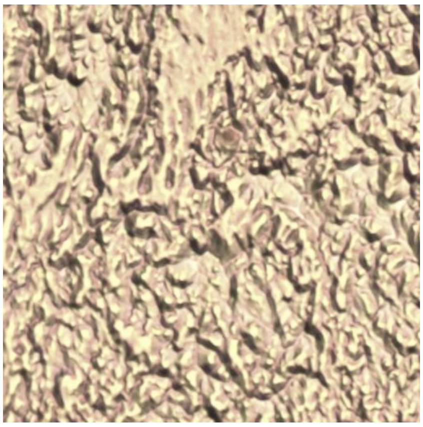

Figure 1. A copper image with 8 ECAP passes.

Preprocessing

Several approaches for preprocessing such as min/max scaling [3], standard scaling [4] and smoothing images using median filter [5] were explored. However, the best submission result was achieved without performing any normalization. As a result, the idea of preprocessing was abandoned.

Data Augmentation

Several approaches like flipping and rotating of images were explored in order to augment the data. However, they did not have any substantial increase in the accuracy. As a result, the idea of data augmentation was abandoned.

Feature Extraction

For feature extraction, Local Binary Pattern (LBP) was used [6, 7]. Even though other approaches such as Histogram of Oriented Gradients (HOG) [8] was explored, LBP was chosen due to its better performance. For LBP, *number of points* and *radius* were chosen to be 8 and 3 respectively. These parameter values were providing the best performance. As a result, from each image 256 feature was extracted. Prior to extract LBP, images were transformed into gray scale.

Feature Selection

Several approaches for features selection was explored such as regularization with L1 penalty, *selectKbest* [9], randomized logistic regression (RLR) [10]. However, these approaches did not result in any gain in the performance. As an example, one explored approach was to select around 20% of features using RLR and using these features with ensemble methods such as extra tree classifier [11] to model the classifier. Even though this approach was performing very well (based on 10-fold classification accuracy results on training data), however, it did not improve classification accuracy of the test results when compared to best submitted accuracy score. As a result, the idea of feature selection was abandoned.

Training

In order to train the best performing model for classifying the copper images different models such as logistic regression (LR) [12], linear discriminant analysis (LDA) [13], support vector machine (SVM) [14], K-nearest neighbor (KNN) [15] and ensemble methods such as random forest, gradient boosting and extremely randomized trees were explored [16]. Each model had fairly good (i.e. above LBP and KNN benchmark) classification accuracy if the parameter were chosen correctly with LBP features. For instance, using LDA with shrinkage could increase the classification accuracy by around 15%. In order to evaluate each model's classification accuracy and choosing the best performing model, 10-fold cross validation was used [17]. In order to choose the best parameter values for each model, grid search approaches was used [18].

Best Solution Model

The best result was achieved by LR with *C* value equal to 0.00001 and default values for other parameters. This resulted in 75% of classification accuracy for the public section of the test data (and ~73% of classification accuracy for the private section of the data). To improve the accuracy, test data was used to for training using the following approach. After first round of classification with LR, prediction probabilities for each sample for the winning class label was extracted and samples and their predicted label having prediction probability above inclusion threshold $\tau_{inclusion}$ (e.g. 70%) was used to for training the model. This increased the classification accuracies to 79.4% and 77.2% for the public and private section of data respectively. More than two rounds of retraining by the test data seemed to not improve the classification accuracies.

Discussion and Conclusion

One observation we made was that even though we were getting sometimes very good 10-fold cross-validated accuracy results on the training set (sometimes even 87% for the median score of 10 folds in a 10-fold cross validation), however, for some submissions the classification accuracy of test data was not satisfactory (i.e. below 70%). These results were achieved even though we were accounting for overfitting and addressing it by for instance performing regularization. We believe that this is an angle that calls for more attention.

In the few last submissions, we tried to use the extra training data we collected using microscope. Each high resolution microscope image was divided into 12 images. This primarily resulted in 360 more images for each class label 1 and 6. However, some of the images were discarded since the camera was out of focus. We trained different models with the training data plus extra data, however, it did not improve the classification accuracy over test data. We observed that many images that were being accurately classified to group 8 previously were being classified to group 6. We believe that this was because of the unbalanced nature of the new training dataset. Even, using stratified k-folds cross validation [19] on the new training dataset did not point to any issue.

With the hindsight, it can be observed that the approach used for our best solution model, was optimizing the model to improve the accuracy of public portion of the data since there is a ~2% difference between public and private accuracy scores in the best solution model. This issue needs to be taken into account and future approaches need to address it.

We were unable to explore other models such as traditional neural networks and convolutional neural networks (CNN) due to time limitations. However, it will be worthwhile to observe what the classification accuracies would be if these approaches will be used. Luckily, after the end of competition, participants are still allowed to do submissions and this provides a good opportunity to explore other promising approaches.

In conclusion, we believe that our model (ranked second in the competition) is a simple yet powerful model while it is fast in both training and prediction and it does not require lots of computational resources.

References

[1]. https://inclass.kaggle.com/c/copper-analysis
[2]. https://inclass.kaggle.com/c/copper-analysis/data
[3]. http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
[4]. http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html
[5]. http://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.median_filter.html
[6]. https://en.wikipedia.org/wiki/Local_binary_patterns
[7]. http://scikit-image.org/docs/dev/auto_examples/plot_local_binary_pattern.html
[8]. https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients
[9]. http://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html
[10]. ttp://scikit-learn.org/stable/modules/generated/sklearn.linear_model.RandomizedLogisticRegression.html
[11]. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html
[12]. https://en.wikipedia.org/wiki/Logistic_regression
[13]. https://en.wikipedia.org/wiki/Linear_discriminant_analysis
[14]. https://en.wikipedia.org/wiki/Support_vector_machine
[15]. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
[16]. http://scikit-learn.org/stable/modules/ensemble.html
[17]. http://scikit-learn.org/stable/modules/generated/sklearn.cross_validation.cross_val_score.html
[18]. http://scikit-learn.org/stable/modules/grid_search.html
[19]. http://scikit-learn.org/stable/modules/generated/sklearn.cross_validation.StratifiedKFold.html