# Report Lab 3

Pattern Recognition NTUA

9th Semester

Efthymios Tzinis 031212007
Konstantinos Kallas 03112057

January 9, 2017

## Introduction

In this work we prepared steps 10 till 17 from the Laboratory Exercise 3. The main purpose of this Exercise includes the implementation of an automatic emotion recognition system which is dedicated to individual emotion classification of all the song files. We took advantage of the previous results of the preparation (feature extraction of MFCC's and also from the application of miremotion() [2]) and used some files of the first laboratory exercise (k-NN and Naive Bayes implementation). For the Steps 12-14 we performed the evaluation in k-NN and Naive Bayes and also performed a PCA analysis of the features working on WEKA. For steps 15-17 we used exclusively WEKA classifiers and our scripts in order to acquire the relevant classification results. After that we compared our selected system for both activation and valence recognition task with an SVM based classifier system and features extracted automatically using Opensmile Toolkit [3]. Our proposed system, based on SVM, outperforms the baseline in both valence and activation recognition with less extracted features. This last step was not demanded by the exercise but it is essential in order to validate if our classification results are good.

## Step 10

Getting the results from the previous steps of preparation we ignore all instances of the data set with average vote equal to 3. These instances were thought to be misleading for the purpose of classification because they are created from divergent votes (e.g votes = (1,3,5)). We excluded the instances who displayed such divergent behavior either in valence or activation related votes. From the total of 412 instances were left only 359 and 355 for activation and valence respectively. Which is an adequate number of instances for the classification purpose.

We proposed a more analytic schema for the statistical validation of our process. this schema is also followed in literature and was recommended by Lab Assistant Nancy Zlatintsi. We investigated the change between the values of Krippendorf's Coefficient for the result data set after excluding the instances of average voting 3. Obviously the coefficient increases but not as much as in the acceptable levels of literature [1], in which requires values above 0.8. After this naive idea implementation we also investigated more sophisticated ways of excluding instances of the data set such as computing the absolute difference of votes and after that excluding those which are under a standardized threshold. But we did not tested on this laboratory exercise because we acknowledge that it not so relevant with the purpose of this exercise. We portend that by increasing the Krippindorf coefficient we will sustain a much more vote coherent and more inter correlated data set with better classification results.

## Step 11

It is limpid that the exercise step was not properly expressed, as 3-fold cross validation does not match with splitting the data in 80% train and 20% test. Nevertheless, we arbitrarily split the data 3 times according to the aforementioned splitting ratios and performed a validation for every possible permutation.

The split was performed by calling random permutation in matlab environment.

## Step 12

The relevant matrix demonstrates the Accuracy and F-Measure for all the permutations of K-NN for values of $K = 2n + 1, n = 0, 1, 2, 3, 4, 5$. We computed these values by using the previous implementation of k-NN we introduced in Laboratory Exercise 1. For further details in choosing these values and in implementation, the reader should check our first laboratory report. The results can be seen in the diagram 1 below, together with the results after the PCA processing.

## Step 13

Same as the previous step we used our previous implementation of Naive Bayes Classifier we introduced in the first Laboratory Exercise. The cofigurations were kept the same with the diagonal matrix of variances initialized in unitary matrix. For further details in choosing these values and in implementation, the reader should check our first laboratory report. The corresponding matrix holds the results of this model. The classification results can be seen below with the results after the PCA.2

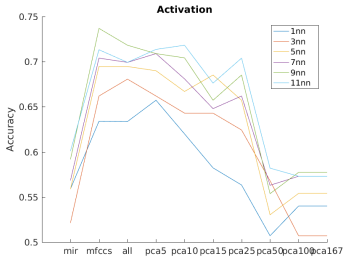
## Step 14

Principal Component Analysis (PCA) is a widely used technique for dimension reduction. We used WEKA in order to perform deviant numbers of principle components. Because of the number of features we portend that the configurations $5, 10, 15, 25, 50, 100, 167$ while calling the weka PCA process will reveal the true nature of the features demanded for a proper classification. In the following diagrams 1, 2 the reader can notice the differences in average F-score and average accuracy. After the process of PCA we re-evaluated our models described in steps 12-13. We can see that only a few number of linear combination of features is actually required for achieving the best classification results. This discloses the true power of dimensionality reduction as much more features do not imply a better classification ratio by any means. The model becomes simpler in computational terms and much more efficient.

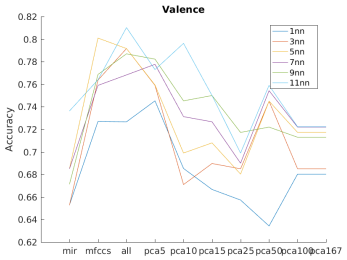
**Important Notice:** For the Steps 12-14 we followed the mixed 3-fold recommendation format of our files but for steps 15-17 we used the auto generated 5-fold mechanism provided internally from WEKA.

## Step 15

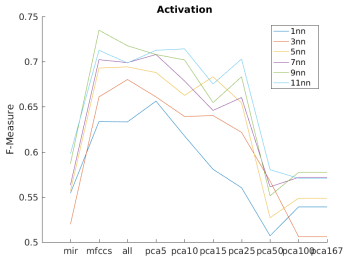
All the classifiers we used for the validation are implemented in WEKA. Thus we used these implementations cause it was more convenient and of course these implementations are much more efficient than ours. All arff files used for this purpose can be found in the corresponding subdirectories of our source code if they are recreated (source/features).



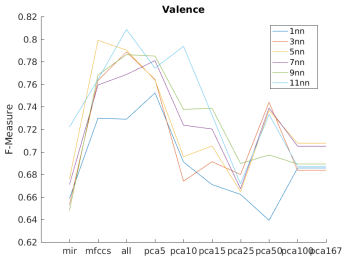

(a) Knn Activation Accuracy




(b) Knn Valence Accuracy

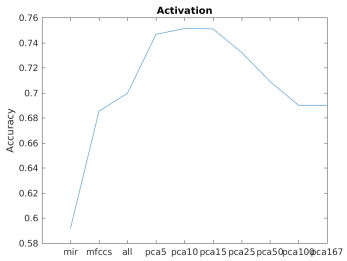



(c) Knn Activation F-Measure

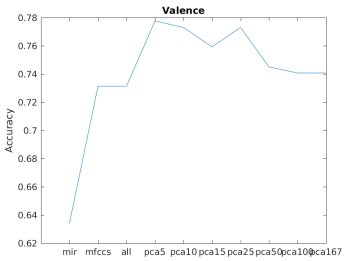



(d) Knn Valence F-Measure

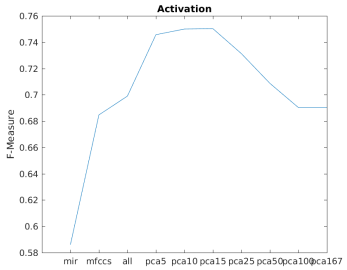
Figure 1: Knn Classification Results



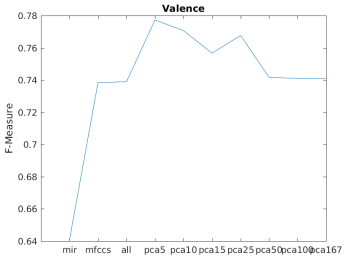

(a) Bayes Activation Accuracy




(b) Bayes Valence Accuracy




(c) Bayes Activation F-Measure




(d) Bayes Valence F-Measure

Figure 2: Bayes Classification Results

## Step 16

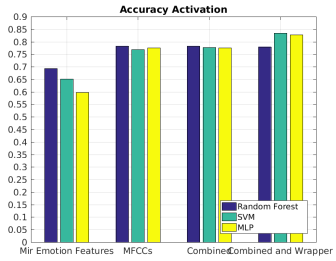
We called the recommended classifiers through bash scripts. For the Multi Layer Perceptron we use a diagram in order to show the results of 5-fold cross

validation process for deviant learning rates and number of hidden layers. The matrix demonstrates all the results of these 3 classifiers. At the end of the report the reader can also examine some graphs showing results from the MLP configuration experimentation 5. (We intentionally did not present all graphs from the experimentation so that the report doesn't grow beyond 100 pages, the reader can see the appendix for all the figures)

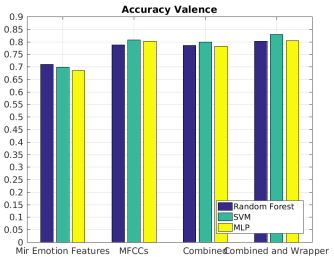
## Step 17

We used WEKA Attribute selection with the Wrapper functionality for both activation and valence in order to extract the most prominent subset of the whole features derived from steps 6-7. We used SVM with the default configuration as an evaluation classifier for the wrapper. We can see that these features are only 10 for activation and 16 for valence. It is useful to mention that the selected features corresponding to miremotion are only 2 and only for valence. All the other features are MFCC based which is something that unravels the superiority of Cepstral features even in a task not closely related to speech recognition. For this set of features we run the 5-fold cross validation models again for all 3 selected classifiers. We chose the best MLP configuration after careful experimentation and evaluation of many different configurations. We also experimented with random forest and svm but the default values gave far better results, so they were used for the final evaluation. Below the reader can see the final evaluation results for the three selected classifiers and for all subsets of features (Mirfeatures, Mfccfeatures, Mir+Mfcc features, Wrapper selected features).

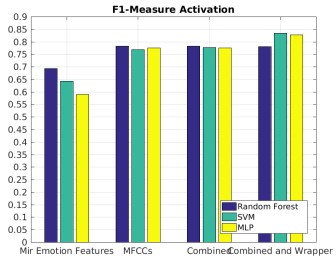
For the selection of Mir emotion Features we found a relevant paper which selects (harmonicity, average energy, tonality, rhythm and scale) [5]. All in all, we employed all the characteristics recommended by the preparation part of this laboratory exercise and we further included all the aforementioned. The final evaluation figures for Random Forest, SVM and MLP classification algorithms can be found in 3. In this table we demonstrate Accuracy and F1-Score for Activation and Valence. Each Subdiagram is comprised by 4 block-bars representing features extracted from Miremotion, the MFCC's, A Concatenation of these features and a feature selection from the concatenation, respectively. Note that in order to deduce the best configuration for MLP, SVM and Random Forest we experimented on many different configurations. For an extended analysis in MLP configuration you can see the Appendix.



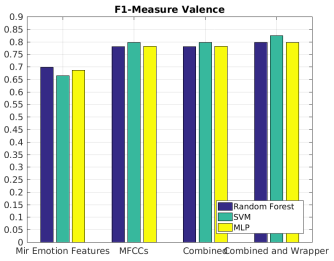

(a) Final Activation Accuracy




(b) Final Valence Accuracy




(c) Final Activation F-Measure




(d) Final Valence F-Measure

Figure 3: Final Evaluation

## Extra Independent Research

In order to compare our system to a renowned and credible co-existing system we tried to extract some features with Opensmile [3] and evaluated them on a different implementation of SVM. The implementation we used is LibSVM [4]. We experimented on different kernels and configurations in LibSVM and in the Tables below 1, 2 we demonstrate the results of all these configurations. Some of them were not working properly like the SVM kernel for the configuration of 3rd degree polynomial kernel.

The configurations Emobase and Emobase2010 are provided from the Opensmile in Linux version and represent a reliable measure in our musical emotion recognition task, as these configurations were employed for emotion recognition tasks with high accuracy. A short presentation of the features extracted for this purpose can be found below:

The feature set specified by **emobase.conf** contains the following low-level descriptors (LLD): Intensity, Loudness, 12 MFCC, Pitch (F0), Probability of voicing, F0 envelope, 8 LSF (Line Spectral Frequencies), Zero-Crossing Rate. Delta regression coefficients are computed from these LLD, and the following functionals are applied to the LLD and the delta coefficients: Max./Min. value and respective relative position within input, range, arithmetic mean, 2 linear regression coefficients and linear and quadratic error, standard deviation, skewness, kurtosis, quartile 13, and 3 inter-quartile ranges.

For the **emobase2010** configuration: The set contains 1 582 features (same as the INTERSPEECH 2010 Paralinguistic Challenge set) which result from a

| Activation Accuracy | Linear | Polynomial 3rd | RBF |
|---|---|---|---|
| Emobase | 75.49 % | 62.67 % | **81.06%** |
| Emobase2010 | 76.88% | 62.67% | 83.01% |

Table 1

| Valence Accuracy | Linear | Polynomial 3rd | RBF |
|---|---|---|---|
| Emobase | 76.9 % | 54.36 % | **82.82%** |
| Emobase2010 | 78.8% | 54.37% | 80% |

Table 2

base of 34 low-level descriptors (LLD) with 34 corresponding delta coef- ficients appended, and 21 functionals applied to each of these 68 LLD contours (1 428 features). In addition, 19 functionals are applied to the 4 pitch-based LLD and their four delta coefficient contours (152 features). Finally the number of pitch onsets (pseudo syllables) and the total duration of the input are appended (2 features). The only difference to the INTERSPEECH 2010 Paralinguistic Challenge set is the normalisation of the maxPos and minPos features which are normalised to the segment length in the present set.

## Acknowledgments

Our model outperforms a renowned emotion baseline offered by Opensmile in recognizing activation and valence from music. Moreover it is perspicuous that our model with feature selection, based on SVM and a polynomial kernel of 3rd degree from WEKA, has the best results. All in all, we compare in the last matrix 3 all the different feature selections. We used the best configuration from OpenSMILE, **emobase.conf**. Maybe more features could be employed through a better classification of the aforementioned activation and valence features, according to the configurations we described before.

## Appendix

The user below can see the images from all the plots of the MLP configuration. We can derive some useful information from all these diagrams. Learnign rate is crucial in some configurations, especially in these ones with only one hidden layer. If the learning rate is not yielding toward 1 then the classification accuracy would be diminished. As we employ more layers in the configuration our non linear classifier enjoys a higher expressivity asset and thus it can describe the emanated patterns more accurately. Activation expiates a more understandable pattern in MLP configurations than the Valence. This is something that

| Accuracy | Emobase Features | Our features | Our features with Wrapper |
|---|---|---|---|
| Activation | 81.06 % | 77.75 % | **83.38%** |
| Valence | 82.82% | 79.94% | **83%** |

Table 3

discloses the surreptitious nature of Valence features throughout a classification problem.

For the creation of all these diagrams we kept all the default values of WEKA except the learning rate and the number of Neurons per Layer. We checked all the configurations but there was no point in plotting all different configurations. Thus we demonstrate only for 3 layers which found to be the best one and we plotted the number of neurons in the final layer alongside the learning rate and in the Z-axis we show the accuracy and the f1 score. We experimented on different number of layers. Consequently, for the 2-level we selected the hidden layer one to be comprised by 15 neurons. For the 3-level configuration we selected 15 and 20 neurons in the previous layers 1,2 respectively. See 5.

In order to provide a user friendly version of our code we provide in this section some concise information related to the nature of every script. The reader is enhanced to re-run all the code for validation and useful feedback.

We provide a Makefile in order to help the user if he wants to re-evaluate our model. He simply has to change the WEKA for working. Although, in every other script (matlab) the user should change the folders of the workspace where is needed. Below we provide a basic description for each code file. The user can run the whole project by just typing **make full_run**. There are others sub-goals in order to provide a fully controllable makefile. (The files we did not provide a description for, are not used in the final project pipeline)

For the final part of our exercise which depends ultimately on OpenSMILE and LibSVM you should change all the paths for Opensmile, LibSVM and Data, manually inside the scripts which can be found in the folder **opensmile_libsvm**.

**lab3_preparation.m** Extracts features from the music files.

**lab3.m** Saves features for use outside of matlab and classifies using naive bayes and knn classifier.

**call_knn.m** Used to call the knn classifier from matlab.

**Compute_Differencial_Metric_Ordinal.m** Computes Krippendorf's Alpha

**final_plots.m** Graphs the final plots with all 3 weka classifiers.

**find_knn.m** Auxiliary function for knn classification.

**find_mfccs.m** Computes all mfcc features from a music piece.

**graph_bayes.m** Graphs the naive bayes results with and without PCA preprocessing.

**graph_knn.m** Graphs the knn results with and without PCA preprocessing.

**make_graphs.m** Calls *graph_bayes.m* and *graph_knn.m*.

**normalize_matrix.m** Normalizes the feature matrix for better classification results.

**call_weka_knn.sh** Calls the knn classifier through Weka.

**call_weka_naive_bayes.sh** Calls the naive bayes classifier through Weka.

**call_weka_random_forest.sh** Calls the random forest classifier through Weka.

**call_weka_svm.sh** Calls the svm classifier through Weka.

**evaluate_for_all_combinations_knn_naive_bayes.py** Gathers all results from knn and naive bayes and prepares them for plotting.

**evaluation_step16_mlp_svm_randomforest.py** Gather svm, mlp and random forest results and calls the final plot function.

**make_an_Arff.py** Converts a standard text file to an arff file.

**mlp_all_configurations.py** Calls the mlp classifier for all configurations.

**mlp_all_configurations_wrapper.py** Calls the mlp classifier for all configurations after wrapper preprocessing.

**mlp_evaluate.py** Gather final mlp results.

**mlp_select_best_configuration.py** Finds and prints the best mlp configuration after experimentation with all configurations.

**pca.sh** Runs the pca preprocessing.

**plot_mlp_plots.sh** Plots the mlp plots with all the configuration combinations.

**prepare_for_weka.sh** Calls the *make_an_Arff.py* script for all requested text feature files.

**txt2svm.py** Prepares features for the opensmile svm classifier.

**wrapper_feature_select.sh** Calls the Weka wrapper feature selection.

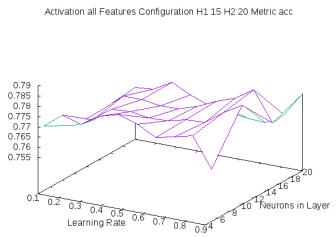
# References


[1] Krippendorff, K. Content analysis: An introduction to its methodology. Thousand Oaks, California: Sage.4 p.241

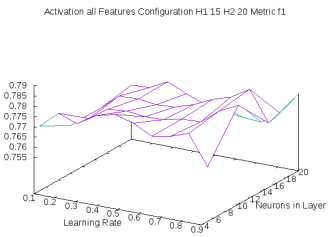
[2] Tuomas Eerola, Olivier Lartillot, Petri Toiviainen, "Prediction of Multidimensional Emotional Ratings in Music From Audio Using Multivariate Regression Models", International Conference on Music Information Retrieval, Kobe, 2009.

[3] Eyben, F., Woellmer, M., & Schuller, B. (2010). openSMILE the Munich open Speech and Music Interpretation by Large Space Extraction toolkit. Retrieved from http://www.mmk.ei.tum.de

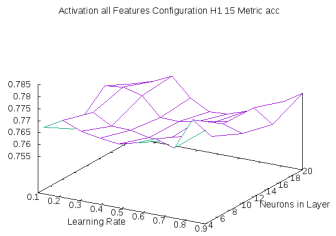
[4] Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines. ACM Transactions on Intelligent Systems and Technology, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm



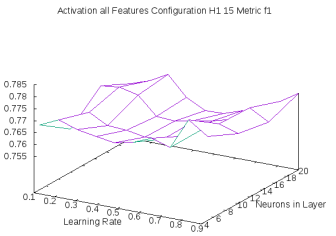

(a) MLP Activation Accuracy 15-20-x neurons



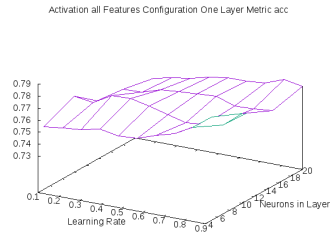

(b) MLP Activation F-Measure 15-20-x neurons



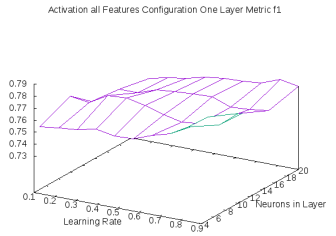

(c) MLP Activation Accuracy 15-x neurons




(d) MLP Activation F-Measure 15-x neurons

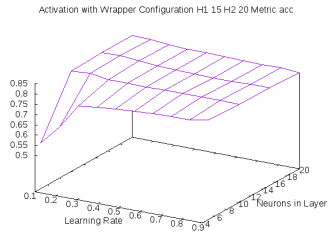



(e) MLP Activation Accuracy x neurons

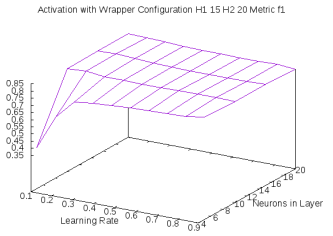



(f) MLP Activation F-Measure x neurons

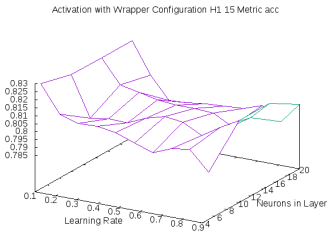
[5] Han, B., Rho, S., Dannenberg, R. B., & Hwang, E. (2009). Smers : Music Emotion Recognition Using Support Vector Regression. In 10th International Society for Music Information Retrieval Conference (ISMIR 2009) (pp. 651656).



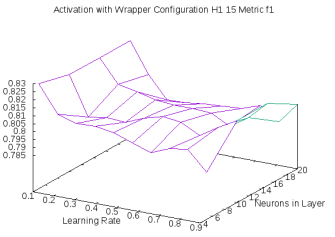

(a) MLP Activation Accuracy 15-20-x neurons (Wrapper)



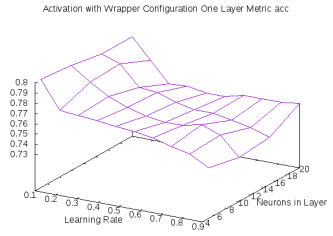

(b) MLP Activation F-Measure 15-20-x neurons (Wrapper)



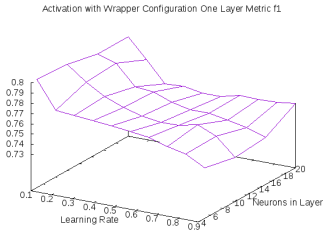

(c) MLP Activation Accuracy 15-x neurons (Wrapper)




(d) MLP Activation F-Measure 15-x neurons (Wrapper)

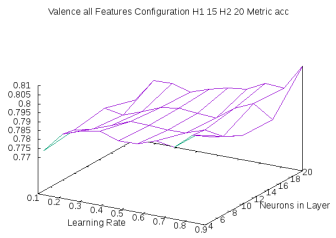



(e) MLP Activation Accuracy x neurons (Wrapper)

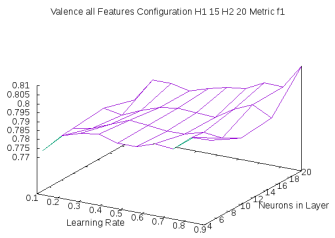



(f) MLP Activation F-Measure x neurons (Wrapper

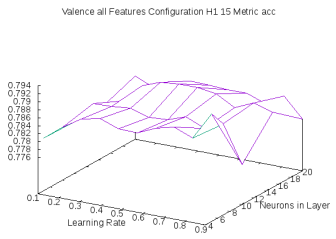
Figure 5



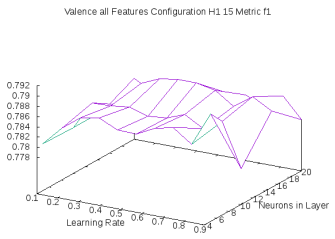

(a) MLP Valence Accuracy 15-20-x neurons



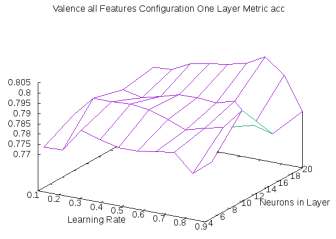

(b) MLP Valence F-Measure 15-20-x neurons



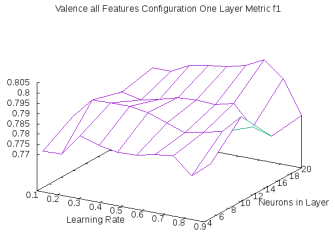

(c) MLP Valence Accuracy 15-x neurons



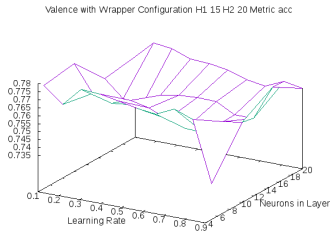

(d) MLP Valence F-Measure 15-x neurons



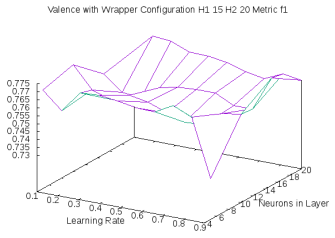

(e) MLP Valence Accuracy x neurons



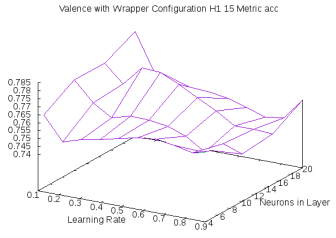

(f) MLP Valence F-Measure x neurons



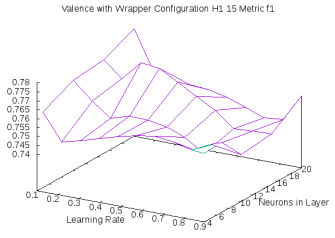

(a) MLP Valence Accuracy 15-20-x neurons (Wrapper)



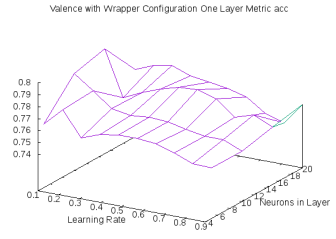

(b) MLP Valence F-Measure 15-20-x neurons (Wrapper)



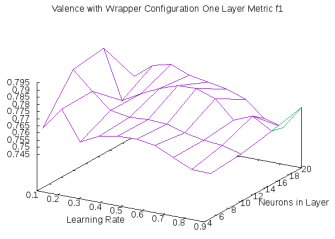

(c) MLP Valence Accuracy 15-x neurons (Wrapper)




(d) MLP Valence F-Measure 15-x neurons (Wrapper)




(e) MLP Valence Accuracy x neurons (Wrapper)




(f) MLP Valence F-Measure x neurons (Wrapper