Active Learning

Intelligent parameter selection for Hardware-in-the-Loop tests

Daniel Haake and Dr. Andreas Maier
The unbelievable Machine Company (*um)

orange™ Business Services

Contents

Advantages of Active Learning 3

Hardware-in-the-Loop (HiL) tests 4

Intelligent parameter selection for HiL tests 5

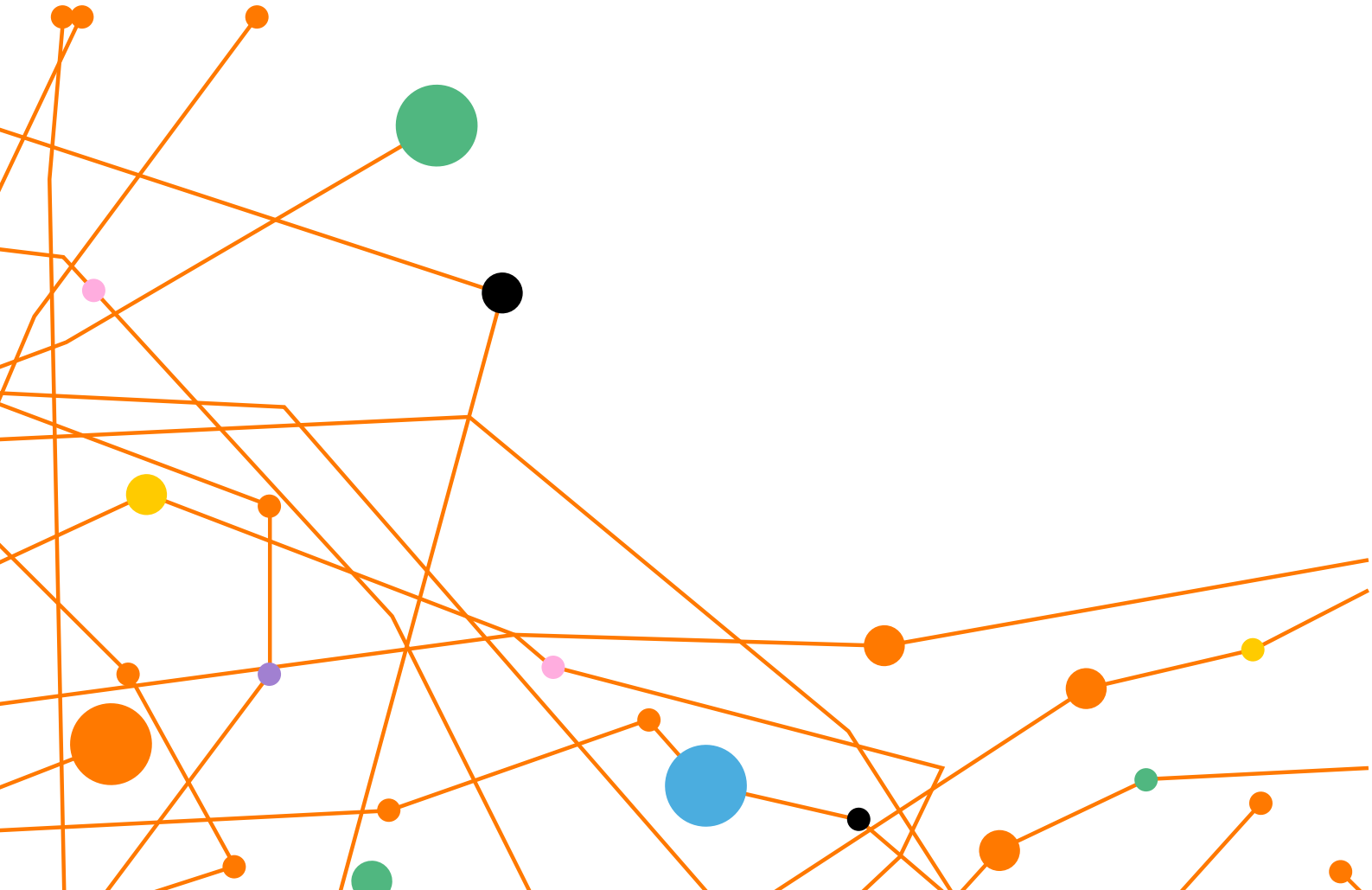
The human parameter 6

Uncertainty calculation 7

Test procedure 8

Determination of success 9

Conclusion 11



Advantages of Active Learning

Active Learning is a form of machine learning used in complex test situations, such as in the functional testing of new components in the automotive industry. Initially, as with machine learning, the algorithm needs to be supplied with training data. However – and this is what is special about it – with Active Learning it can request the relevant training data and suitable parameters itself, and then go on to select them independently once it has the information it needs.

In this way, Active Learning can reduce the amount of training data required and, consequently, the time and cost of entire processes by a factor of ten – therefore delivering results ten times faster than traditional approaches.

Daniel Haake and Dr. Andreas Maier, Senior Data Scientists at The unbelievable Machine Company (*um), have developed and successfully applied Active Learning to hardware-in-the-loop testing of components for a major European automotive manufacturer.

In this white paper, they discuss how they went about it, and the benefits and opportunities afforded by Active Learning.

Hardware-in-the-Loop (HiL) tests

Before new components such as collision warning systems can be installed in vehicles and then sold, they must be tested to ensure that they function properly. One option is to install the vehicle component in a test vehicle and test the component manually. This involves considerable effort, as well as costing time and money. For example, the driving situation needs to be simulated using different initial situations. In addition, different driving speeds must be factored in, as must the distances between the vehicles. Depending on the type of component, there may be several parameters to take on board.

If we imagine a situation in which we only need to consider three parameters, for which we want to test 10 different levels each, this results in $10^3 = 1000$ different initial situations. If further parameters are added, the number of initial situations to be tested is multiplied accordingly.

To get to grips with this problem, the automotive industry uses “hardware-in-the-loop tests”, or HiL tests for short. In these tests, the real world is simulated by a computer. The component to be tested is connected to the computer or HiL simulator, which simulates the vehicle to be driven, via its interfaces. The component’s inputs are then fed with sensor data from the HiL simulator, which makes it possible to test the various parameters directly, one after the other.

Besides the advantage that the driving situation does not always have to be performed repeatedly in the real world, another advantage is that as soon as one test run has been completed, the next one can be started immediately. Furthermore, once a fault has been corrected on the component, the test can simply be repeated to see whether the fault has been corrected.

However, these HiL tests run in real time and iteratively, which makes it impossible to speed up the tests by using faster hardware or parallelization. A potential problem here is that tests can be carried out in situations where the result is already known beforehand, thus making the tests superfluous.

By way of example, let’s consider a test for a parking assistant. If the vehicle is 4.50 m long and the parking space is 4.70 m wide and the parking maneuver fails, then there is no need to test any parking spaces measuring 4.65 m, 4.60 m, 4.55 m or even less than 4.50 m.

For humans, it is only logical not to proceed any further with these tests. For the HiL simulator, however, these are only meaningless numbers; it tests every value of the defined parameter space. Superfluous testing costs time and money and blocks the test bench for other tests.

Intelligent parameter selection for HiL tests

Is it possible to select the test parameters more intelligently, in order to save time and money, yet still achieve high test quality?

It is indeed. One option is to use machine learning and a certain previously defined number of parameters to train a classifier, such as a support vector machine or random forest, for example. A prediction can now be made for the parameters not used in the training, and the predictions can be accepted as test results. But how good are these results really? How do we know that the classifier has been trained using appropriate parameter combinations that provide a good separating plane between the individual classes?

If too few parameter combinations are selected for the training, a random selection of the parameter combinations could lead to those from a certain parameter space being selected more often. Figure 1 expresses this more clearly. Here we see a two-class problem, which slightly overlaps in the middle. There is clearly a good separating plane in the range x = 0. However, when only a few random parameter combinations are selected to train a classification model, the separation line may slant to the left.

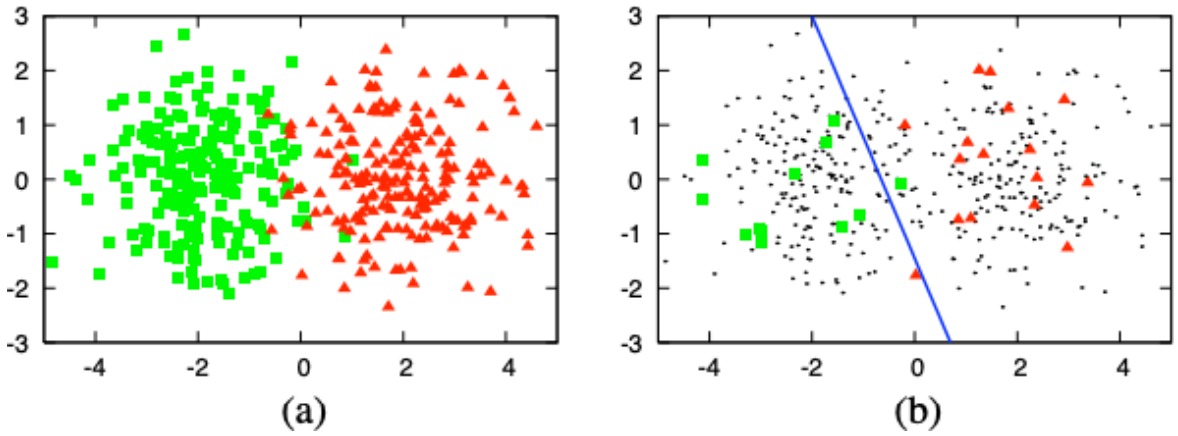


Figure 1: Two-class problem and the problem of incorrect classification when using a random selection of only a few parameter combinations

To prevent this, a large number of parameter combinations must be selected. This corresponds to the law of large numbers and minimizes any shifts caused by randomness.

The best solution would be to divide the parameter space into a training set, a validation set and a test set, as is usual with machine learning problems. But for this we need the class membership for all parameter combinations, which we would only get following a test in the HiL simulator. Here again, each parameter combination of the entire parameter space would have to be tested in the HiL simulator, which would make the use of machine learning obsolete. Moreover, we would only have trained a model that would work for future predictions. Since all parameter combinations have been tested by the HiL simulator, no further predictions would be necessary. In addition, the model could not be used for future test cases since some of them differ considerably.

The human parameter

As human beings, how would we try to find a well-functioning model with just a few data points?

We would first try to develop a model with a few data points and then work to further improve the model over time. How can the model be further improved over time? By going on to test the data points, thereby determining their class membership, which enables us to expect the greatest possible information gain. Let's stay with the example from Figure 1. Let's assume that we have already trained a model with a few data points and that we therefore know that the separation line between the two classes must be somewhere in the middle of the graph. Consequently, we are already quite sure that the two edges of the graph belong to either one or the other class.

So, if we take a data point in the range of x = -4, we are quite sure that the data point will belong to the green class, and in the range x = 4 to the red class. In the range x = 0, however, there is a significant lack of certainty. So now, to improve our model, we would not select a data point to determine the actual class from the range x = -4 or x = 4, but from the range x = 0; this is how we can expect the greatest possible information gain to help us improve on our previous model.

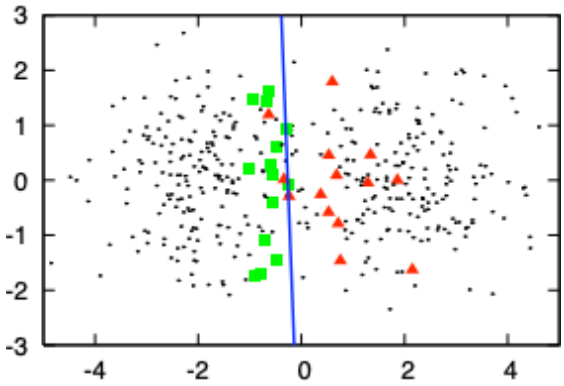


Figure 2: Targeted selection of parameter combinations to improve the classification result (Active Learning)

This concept is in line with the Active Learning method. The basic assumption here is that the model can be improved at an early stage through the targeted selection of parameter combinations, which means that better forecast results can be achieved with fewer data points (cf. Figure 2).

The procedure for Active Learning corresponds to the above assumption of how a human being approaches the problem.

First, a certain number of data points are used to train an initial model. Then, using the computed model, the probability of belonging to a specific class is calculated for all unused data points. The data point(s) for which the model has calculated the greatest uncertainty regarding class membership are used to determine the actual class membership. Once the class membership of the data points with the greatest uncertainty is known, the model is re-trained with the initial data points and the data points with the greatest uncertainty. The procedure is repeated until a defined termination criterion has been reached. This offers the possibility to determine the average certainty of the model as regards the class membership of all data points. This criterion is known as Confidence.

Uncertainty calculation

When determining a data point based on the greatest uncertainty, there are several ways to calculate the uncertainty. One way is to determine what is known as Least Confidence. This involves selecting the data point for which the model has calculated the least probability for the most likely class. If the model is 80 percent certain of its specific class prediction for a data point A, but only 60 percent certain for another data point B, data point B would be selected next.

For a classification problem with more than two classes, a Margin Sampling strategy can also be used. This involves taking each individual class and calculating the probability that the data point belongs to it. The data point selected will be the one where the difference between the two most probable classes is the least. Let's suppose that for a data point A, the probabilities of belonging to a class in a three-class problem are 50% to 49% to 1%, and for a data point B, the probabilities are 40% to 30% to 30%. The difference between the two most likely classes is now 1 percentage point for data point A, and 10 percentage points for data point B. Here, data point A would be selected next. With the Least Confidence method, however, it would have been data point B.

A third possibility is Entropy Sampling. Entropy is a term used in information theory to describe how high the information content of a message is. The entropy of a data point is calculated via the class membership probabilities of all classes K as follows:

$$H_A = -\sum_{i=1}^{k} p_i \log_2 p_i$$

where k is the number of classes and pi is the probability of the data point belonging to class i.
$0 * log2\ 0$ will thus also be defined as 0 on the basis of $\lim_{p_i \to 0} p_i \log_2 p_i = 0.$

Knowing that a data point belongs 100% to class A and therefore 0% to class B gives me no further information. The result is known, the information content is 0. With a 50% to 50% distribution, however, the result provides me with a high information value. The information content is high. This is also reflected in the above formula. For a 100% to 0% distribution, the result is $-(1 * log_2 1 + 0 * log_2 0) = -(0 + 0) = 0$. For a 50% to 50% distribution, the result is $-(0{,}5 * log_2 0{,}5 + 0{,}5 * log_2 0{,}5) = -(-0{,}5 - 0{,}5) = 1$. The entropy value range is thus between 0 and 1. In Entropy Sampling, the data point with the highest information content should be selected. Accordingly, the data point with the highest entropy is selected next.

Test procedure

As described above, the Active Learning algorithm is applied until a termination criterion is reached. This can be a certain number of runs. However, it is also possible to determine how certain the model is that the data points belong to a certain class. This value is known as Confidence and is calculated from the difference in probability of the two most probable classes of a data point.

If the probability for a data point for class A is 97.5% and for class B 2.5%, the Confidence for this data point is 0.95. The average confidence value for all the data points is now considered, in order to determine how certain the model is of its predictions. A specific Confidence value can now be defined for the termination of the Active Learning algorithm.

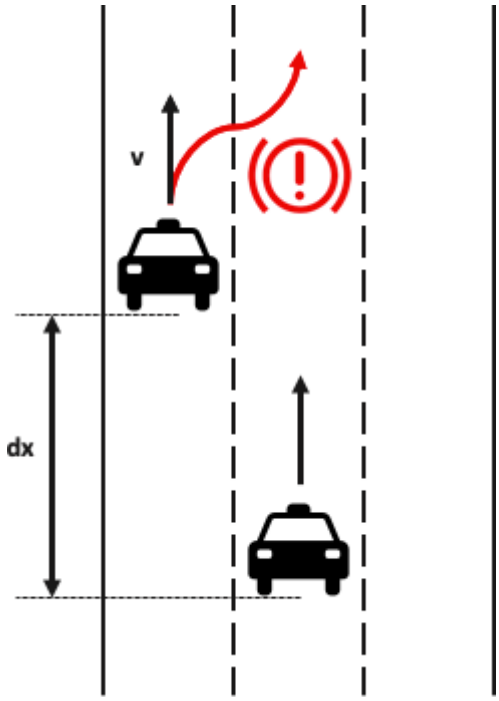


Figure 3: Test scenario for the HiL test

To check how well Active Learning works in HiL tests, data for the following test scenario were available (cf. Figure 3). Driving on a straight, three-lane highway was simulated. The test vehicle drove at a constant speed in the middle lane. The test vehicle's speed was constant for all test scenarios. The so-called co-vehicle is driving in the left lane. This vehicle suddenly swerves into the middle lane and immediately performs an emergency braking maneuver. The test vehicle's collision warning system is being tested here. If the collision warning system issues an audio warning, the test is deemed to have been passed; if it does not, the test is deemed to have been failed. This is therefore a 2-class problem involving the classes "passed" and "failed". The parameters that were variable were the distance between the two vehicles and the speed of the co-vehicle at the start of the test. The two parameters were tested at fixed distances, giving a parameter grid dimension of 22 * 22 (cf. Figure 4).

The red data points signify that the collision warning system has not triggered, while the very faint red data points signify that the collision warning system has triggered. A clear-cut decision boundary can be seen, with just a few outliers.

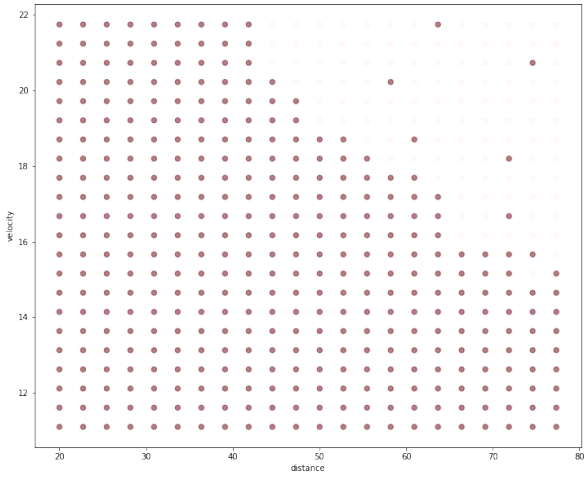


Figure 4: HiL test data

To artificially increase the data base, the parameter grid was increased to 100 * 100 and the missing intermediate values were interpolated on the basis the known classes. In order to make the clear-cut decision boundary somewhat harder to find for the algorithm, an additional artificial five percent noise was generated (cf. Figure 5).

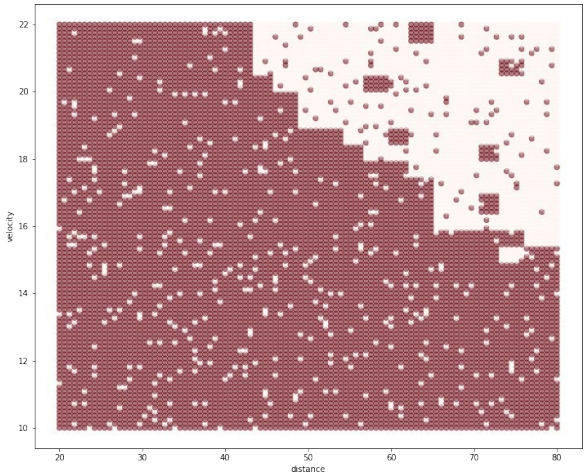


Figure 5: Interpolated data with artificially generated five percent noise

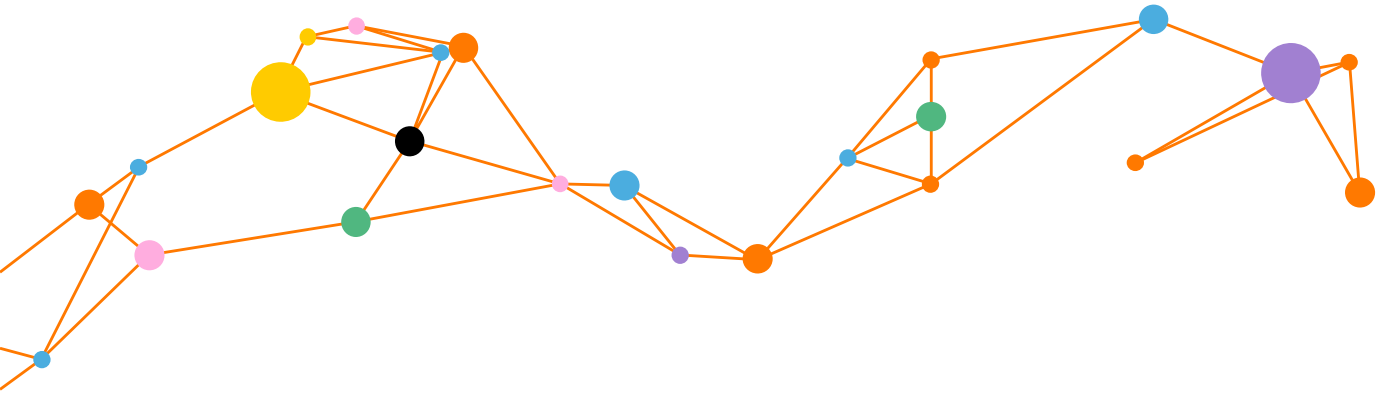
Determination of success

To determine how well the Active Learning algorithm performs, the following procedure was adopted. First, the data set was divided up into a training set and a test set. The training set was used to train the algorithm and the test set was used to calculate how good the algorithm's predictions were, with Accuracy chosen as the unit of measurement. A small number of data points were randomly selected from the training set. These data points were stored in the so-called "Labeled Pool" and were used to perform the initial training. The remaining data points from the training set ended up in the "Unlabeled Pool".

The initial training used an Extra Tree classifier, which is a special type of Random Forest. There are practical reasons behind the choice of classifier. Essentially, the aim is for the procedure to be subsequently used to select the parameter combinations to be tested in a HiL test. Consequently, it makes sense to use a model that is already quite stable against overfitting without hyperparameter tuning.

Using the initial model, the class membership probabilities of the data points in the Unlabeled Pool were calculated. The data point with the lowest confidence was selected, added to the Labeled Pool and removed from the Unlabeled Pool, using the criterion of Least Confidence. The model was then trained with the data points from the Labeled Pool and the procedure, involving selecting a new data point and retraining the model, was repeated 500 times.

The entire procedure was performed again. However, after the initial training, a data point was not selected from the Unlabeled Pool actively, but randomly. This makes it possible to show how the active selection of a data point influences the results as compared to random selection.



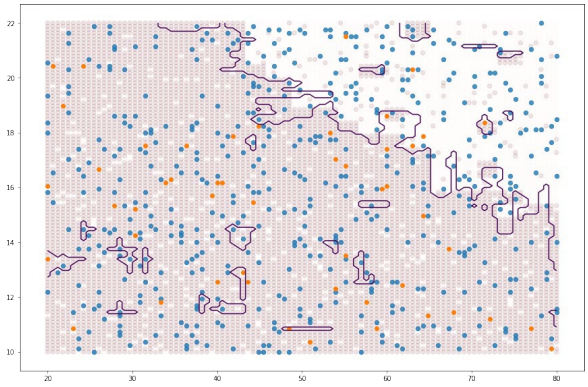


Figure 6: Decision boundary after 500 runs of randomly selecting data points to improve the previously trained model

Figure 6 shows how the algorithm proceeded when additional data points were randomly selected. The orange dots represent the data points that were used for the initial training. The blue dots are the data points that were used to improve the previously trained model. In the case of Figure 6, a new data point was always drawn randomly. In total, the selection of a new data point followed by retraining of the model was performed 500 times. As can be seen, the correct decision boundary was found to some extent, but it is quite unstable and shows a certain degree of overfitting. It can also be seen that the algorithm has allowed itself to be irritated by the noise, which has caused misclassifications to occur in the red class region.

Figure 7, on the other hand, shows how the algorithm, using Active Learning after the initial training, selectively uses the data points that are at the actual decision boundary to improve the model. In the red class region, data points were initially provisionally selected through the noise in order to improve the model. However, only a few data points were selected there until it learned that it was just noise and that the actual decision boundary was somewhere else. Once the data point selection and retraining procedure had been repeated 500 times, the correct decision boundary was located quite clearly, in contrast to what happened with random data point selection.

Figure 7: Decision boundary after 500 runs of actively selecting data points to improve the previously trained model

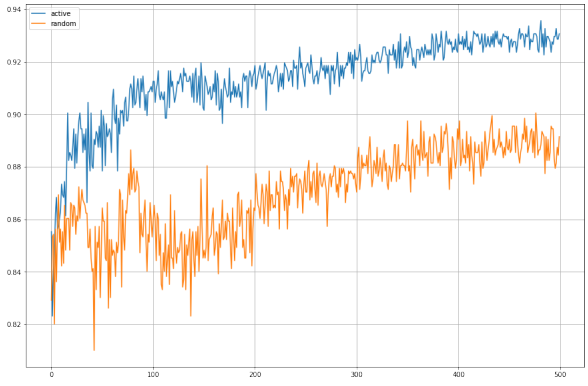


Figure 8: Accuracy development during data selection and retraining of the model, orange: random data point selection; blue: active data point selection

If we now look at the accuracy measurements for the test set during retraining and compare the values for random selection of the data points used in retraining with those for active selection, we find that after 500 runs of retraining, accuracy is significantly greater with active selection than with random selection (0.93 to 0.88 - 0.89, cf. Figure 8).

It can also be seen that accuracy increases considerably faster. The level of accuracy that was achieved after 500 runs of retraining with random selection was already reached after 40 to 50 runs with active selection, thus making it around 10 times faster.

A similar picture emerges when looking at confidence, i.e., the value relating to how certain the model is of its classification of data points (cf. Figure 9). Here too, after 500 runs of retraining, a significantly higher confidence value is achieved with active selection of data points than with random selection (0.95 to 0.82). In addition, the confidence value achieved after 500 runs of retraining with random selection is already reached after 40 to 50 runs with active selection, thus making it also around 10 times faster. Furthermore, as can be seen, the fluctuations in the curve are significantly smaller when the data points are selected actively.

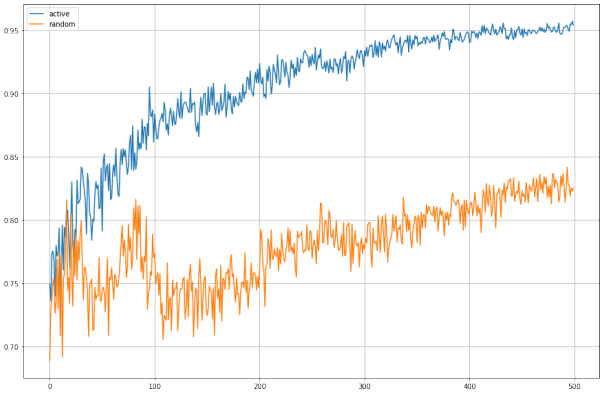
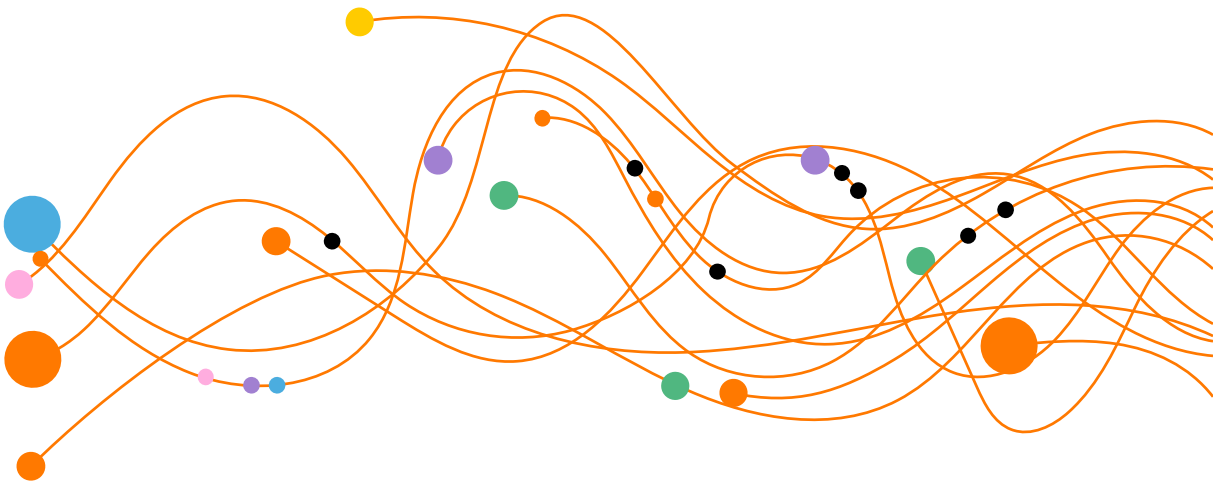


Figure 9: Confidence development during data selection and retraining of the model, orange: random data point selection; blue: active data point selection

However, a specific problem must be noted: if we take a closer look at Figure 7, we can see that no islands were found in the upper righthand corner when selection was performed actively. The Active Learning algorithm does not search there because it has no knowledge of the fact that there could be islands in that region. To alleviate this problem, the parameter space should be divided into several regions before the initial training is performed. The data points for the initial training should not be drawn at random, but should instead come from each region of the parameter space. Also, when data points are selected actively and a certain number of runs have been performed, a data point should either be selected randomly or come from a parameter space region that has not yet been tested. In this way, a finer subdivision of the parameter space can be achieved compared to that available before the initial training, and it can also be defined increasingly finely over time. This can ensure a larger space is covered as well as helping to locate islands.

Conclusion

In summary, we can see that the Active Learning approach is very well suited to achieving high prediction quality even with just a few data points. Active Learning is particularly useful for HiL tests, since its intelligent selection of test parameters avoids unnecessary tests, thus offering considerable savings in terms of time and money.



Authors

Daniel Haake has a B.Sc. in Computer Science / E-Administration, an M.Sc. in Data Science and was Senior Data Scientist at The unbelievable Machine Company. He received the 2020 Future Award for Police Work for his Master's thesis on "Predicting Residential Burglaries with the Help of Machine Learning Algorithms".

Dr. Andreas Maier holds a PhD in physics and is a Senior Data Scientist at The unbelievable Machine Company. Previously, he worked as a Software and Big Data Engineer for the Leibniz Computing Center (LRZ) in Garching, Axel Springer Ideas GmbH and Zimory GmbH, among others.

The unbelievable Machine Company (*um)

The unbelievable Machine Company (*um) is the leading full-service provider for ambitious digital projects. Specializing in Big Data (Data Science and Data Engineering) and Cloud Services, the company develops precisely tailored solutions to the unique business challenges facing big-name customers in the automotive, retail, online services and media industries, as well as many other sectors. As part of the Basefarm Group (an Orange Business Services company), *um offers high-quality local services through its global, Europe-based infrastructure. It has already received numerous awards and was most recently named "Data Analytics Leader" by ISG / Experton for the fifth time in a row. Founded in 2008, today around 150 incredible employees work for *um at its Berlin, Frankfurt and Vienna locations.

Contact

The unbelievable Machine Company GmbH
Michelle Zirnsak (Marketing & Sales)
Grolmanstr. 40
D-10623 Berlin
+49 (0) 30 88926560

marketing@unbelievable-machine.com
www.unbelievable-machine.com

orange™ Business Services