TPOT's performance for Biomedical Data

A Data Mining Seminar

T.P.A. Beishuizen (0791613)
Biomedical Engineering - Computational Biology
Data Engineering - Information Systems
Eindhoven, University of Technology
Email: t.p.a.beishuizen@student.tue.nl

February 9, 2018

Contents

1 Introduction

At the Computational Biology department (cBio) of Biomedical Engineering (BME), many requests are made to analyse gathered data. This data usually stems from research in hospitals, but can also be from other BME groups and publicly available data. Currently a standard is missing to efficiently analyse those data sets. With the vast number of data sets that are available, such a standard in the form of a framework on data analysis would be valuable. It would speed up projects and give them a higher chance to succeed the goal, due to improved efficiency.

An example of biomedical data sets was based around gene expression of skin diseases. Two skin diseases were tested, psoriasis and atopic dermatitis, the latter one better known as a form of eczema. The expression of a big number of genes was tested for skin disease patients on skin affected by the disease (lesional skin) and skin not affected by the disease (non-lesional skin). At last there were normal patients, that did not suffer from the skin disease. Nine data sets were available for these data set, six for psoriasis and three for atopic dermatitis. The number of tested skin plaques ranged from 28 to 180 whereas the number of tested genes is the same for every set, namely 54675.

A possible solution for partly providing a framework for cBio can be found in automated machine learning (autoML). This relatively new extension to machine learning tries to automatically search the best combination of preprocessing, feature selection and machine learning algorithms to efficiently describe a data set. CPU or actual time and memory usage are the two main constraints for autoML, due to testing different methods with varying parameters taking up time and space. These autoML algorithms usually are made with the combined algorithm selection and hyperparameter optimization (CASH) challenge in mind. [1]

Olson and Moore proposed a tool for autoML, a Tree-based Pipeline Optimization Tool (TPOT). [2] This tool also tries to find the best classifier by creating pipelines for the different algorithms. The pipelines are evaluated and branched or altered according to the evaluation. It makes use of genetic programming, a technique used for evolutionary computations. [3] Evaluation of several TPOT approaches are proven better than doing a basic machine learning analysis and are therefore a promising approach for implementation. [4] Several projects used layering and meta-learning techniques to tackle the time and space constraints. [5]

To test if TPOT is also suitable for biomedical data, it must be tested by a specific set, such as the bariatric data set. This data consists of several biomedical data challenges only some of which seem solvable by TPOT. A research must be done to find out whether TPOT can manage to retrieve good results from this data set and may benefit of some improvements.

For the data seminar, first the background will be given. This background will be about the biomedical data sets, the exemplary data set used for the seminar, a topic of automated machine learning and at last a part about TPOT. Secondly the research question is given, together with a hypothesis.

2 Background

2.1 Biomedical data

Biomedical engineering can be seen as a specific part of engineering with a wide variety of topics. These topics can be theoretical, non-experimental undertakings, but also state-of-the-art applications. Not only research and development can be used, but also implementation and operation. Combining all of these different parts in one definition is hard. [6] For this project, the focus is mainly on research and development, also known as knowledge discovery. [7]

When a biomedical engineer starts a project, at the start usually only a data set and the research goal are known. To achieve that certain goal from the data set, four aspects influence the project's course and development. At first obviously the data itself is a big part of such an influencer as the research is restricted to limitations from it. Examples of such restrictions are multidimensionality, set size, data heterogeneity, missing feature values and population handling.

The other obvious influencer is the main research goal. Since the biomedical engineer wants to achieve a certain goal, the approach outcome must match that goal for the research to be successful. Most goals are focused around either data mining, extracting relations from available data, or modelling, creating a model within data features. A third influencer is the availability of data analysis tools. The steps to take from data to goal do not only include an approach, but also a tool to execute it. The choice of a certain tool has a big impact on the project, as each one of them has its own advantages and disadvantages. The two most well known tools within BME are MATLAB and Python, however some engineers have used R, Java or C++ and there are still other possibilities. A last big influencer is the biomedical knowledge. What experience the scientist already has with similar projects can greatly influence the choice of approach and framework. Knowledge of the supervisor and publicly known information on the research subject from books and articles also influence the approach, as already known outcomes do not have to be researched again.

For data engineers the main focus lies in trying to find patterns in the data. Therefore in this seminar the focus will mainly be on the data driven aspect of a biomedical project. Characteristics of such a data driven approach (as mentioned before) are mainly focused around data volume, dimensionality, complexity, heterogeneity and quality. [8, 9]

Collecting data because it is possible can make data sets bigger than needed. Both in number of instances and features, data sets can be harder to understand or analyse when more is available. [8] This volume problem usually is tackled by taking sub-populations of the complete set. These sub-sets can either be focused around a part of the population (gender, age, race) or taken at random to still represent all of it. Due to the efficiency of analysis techniques and the rise in computational speed of servers [10], volume on its own becomes less of an issue. Volume does however become an issue when combining with heterogeneity and quality. [11, 12]

Not all data sets have a high number of instances that cause a big data volume. Sometimes there are relatively few instances, while the number of features is proportionally high. [13] Usually many of those features are not relevant enough for the research, however are still used for testing. Trying to remove features that are not important, will greatly help finding relations between the others and create more knowledge about the research topic. Lowering the number of features also makes the data volume go down, so analysis should be easier. Mainly an optimal features set should be selected to obtain the best results. [14]

Biomedical data can also be very complex. Useful results may be present, however it can be very hard to obtain it. Examples of complex data are images, several biomedical signals and temporal data. Details of the useful results that are present in images can for example be very hard to detect, the temporal data can vary quite much over time and the biomedical signals can be hard to combine with static biomarkers. [15] This aspect can benefit from exchanging knowledge with other research areas that specialize in mining of those complex data sets. [11, 16]

The biggest challenge encompasses aligning different data sets. No standard for data sets is available and therefore data sets differ greatly from each other. Data is weakly structured or even unstructured [12] and variables are processed differently due to other protocols or the collectors' preference of representation. [17] Also the variety of data is hard to combine when sources are fundamentally different. When parts of the data are images, another part is a table from the laboratory and a third part is textual remarks of the doctor, standardizing merging those three is much harder than merging three lab sets. Those merges are also very prone to errors, as imprecisions can be vastly different between those data sets. No tool can work directly with these raw data sets and preprocessing must almost definitely occur beforehand. [11, 18]

A last challenge is about data quality. The data is usually gathered by doctors and laboratory workers. Since the data is manually gathered by humans, the data have a relatively high error rate. Therefore the data can be quite noisy, values can be inconsistent, wrongly entered or even missing. [18] Not only human errors cause the data quality to drop, but the heterogeneity, as well. Two hospitals might have different protocols for the same treatment and sample different biomarkers for that protocol. Due to that difference, biomarkers may be missing for some of the entries. The time of data gathering is also a big factor as some biomarkers change greatly over time. The databases are usually also built for financial purposes and not for research, which can

hurt the quality. [15]

These challenges within the data are greatly discussed. [16] Many proposals to tackle them are made, however none is actually widely adopted, yet, as a global standard for databases. Also, with the uncontrolled growth in biomedical data, it will become hard to have such a standard recognised all over the world. [17, 19, 20, 21, 22, 23]

2.2 Skin Diseases Data sets

Skin diseases could form a major disability in someone's life. Whereas skin diseases were not as life threatening as diseases such as Cancer, Alzheimer and AIDS, the could lower quality of life significantly. When looking at health-related quality of life (HRQL), patients with psoriasis showed same problems as patients with other major chronic health conditions. [24] Patients with both psoriasis and eczema suffered from severe itching symptoms and possibly even severe pains. Further insights in these skin diseases could help alleviate their unwanted side-effects and help improve the patients' quality of life. [25]

Information on both of these skin diseases could be found from nine data sets. The data sets consisted of information on skin of patients with the disease (lesional skin) and skin without the disease (non-lesional skin). In several experiments this skin was taken from the same patient. Also some skin was taken from patients not suffering from the diseases at all. Six data sets focused on Psoriasis and three focused on atopic dermatitis. These data sets consisted of a total number of 54675 features, each of them corresponding to a specific gene. Not many skin samples were taken, ranging from 28 to 180. Also, since every data set was created by different people, some minor differences were present in them as well (Table 1).

The nine data sets are rich in information. The dimensionality is very high and if combined also houses a decent number of samples. Several challenges arise in the data set, too, as biomedical data sets often have (subsection 2.1). Three of these challenges are discussed for this case.

At first the challenge of handling nine different data sets was important. Even though the sets were created based on the NCBI database [34], the layouts were not identical. These difference originated from the intended research goals and the data availability. It is not possible to just concatenate samples without some form of preprocessing. Only the parts that are the same all over the data sets should be taken and all other parts omitted. A first look would be best on the nine data sets separately so initial ideas found with as less bias from combining them as possible.

A second challenge could be found in the high number of features. There were 74675 featured measured, averaged a 1000 times the number of samples. The genes that actually were significantly involved in the skin diseases however should be about $1/1000^{th}$ of the total number of measured genes. Many features should be redundant and removed during preprocessing, a valuable and complex step in biomedical data mining.

The third challenge was about data volume. The number of samples differed from 28 to 180, all of them being a very low number compared with the number of features. This indicates that the number of samples represent the complete sample space poorly, not clearly showing the boundaries between the areas. This could create problems during machine learning with such a low training and test set. Several cases will arise where accidentally all training and test set agree with the algorithm, whereas other samples from the sample space would not.

2.3 Automated Machine Learning

Before automated machine learning (autoML) existed, a dataset was mined by hand. First a preprocessing algorithm was chosen and used to prepare the data. Next a (machine learning) algorithm was chosen to mine the desired results out of the data. At last the hyperparameters of the chosen algorithm were tuned to optimize the desired results. These three steps are vastly different and significant issues arise when combining these. Several ideas arose to combine the steps, called Combined Algorithm Selection and Hyperparameter optimization (CASH). [35] After some time, when preprocessing was added in the mix as well, the name autoML was being used. [5] The first autoML approach tool was published as `Auto-WEKA` that focused on classification methods,

Table 1: Details of the nine skin disease data sets. The number of samples and features has been given, as well as remarks of the skin types.

Disease	Data set name	Sample size	Features	Remarks
Psoriasis	**GSE13355** [26]	180	54676	Three skin types: - NN (normal, 64 samples) - PN (non-lesional, 58 samples) - PP (lesional, 58 samples)
	GSE30999 [27]	170	54676	- No normal patients - Non-lesional (85 samples) - Lesional (85 samples)
	GSE34248 [28]	28	54676	- No normal patients - Non-lesional (14 samples) - Lesional (14 samples)
	GSE41662 [28]	48	54676	- No normal patients - Non-lesional (24 samples) - Lesional (24 samples)
	GSE78097 [29]	33	54676	Different types of skin samples: - Normal (6 samples) - Mild Psoriasis (14 samples) - Severe Psoriasis (13 samples)
	GSE14905 [30]	82	54676	- Normal skin (21 samples), - Non-lesional skin (28 samples) - Lesional skin (33 samples)
Atopic Dermatitis	**GSE32924** [31]	33	54676	- Normal skin (8 samples) - Non-lesional skin (12 samples) - Lesional skin (13 samples)
	GSE27887 [32]	35	54676	Different type of skin samples, pre and post treatment of skin: - Pre non-lesional (8 samples) - Post non-lesional (9 samples) - Pre lesional (9 samples) - Post lesional (9 samples)
	GSE36842 [33]	39	54676	Also difference between acute and chronic dermatitis. - Normal (15 samples) - Non-lesional (8 samples) - Acute lesional (8 samples) - Chronic lesional (8 samples)

spanning 2 ensemble methods, 10 meta-methods, 27 base classifiers and their hyperparameter settings. [35] An upgrade was published that added regression and parallellism. [36]

As explained shortly before, to go from data and results several steps must be taken: Preprocessing, algorithm selection and hyperparameter optimization. This sequencing is called a machine learning pipeline. Such a pipeline can consists of zero, one or multiple preprocessing steps for data preparation, can be one of many different machine learning algorithms which on their part have wide ranges for multiple hyperparameters. The explosion of possible pipelines makes it hard to choose the right one. Knowing successful combinations is useful, however every data set has different features that ask for different pipelines. [5]

AutoML tires to find the best machine learning pipelines to compute which algorithms must be selected, combined with tuning the hyperparameters and preprocessing. This algorithm selection usually is done in a meta-learning approach, which focuses on finding how the machine learning algorithms perform for a task interval. Hyperparameter optimization has challenges on his own to

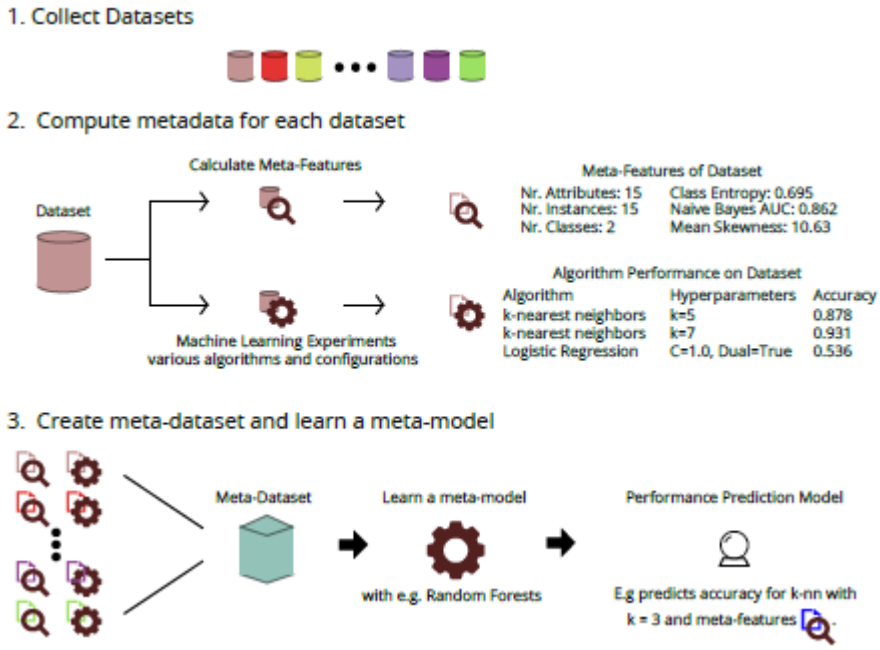


Figure 1: A layout of how meta-learning works. 1. Data sets are collected. 2. Meta-data is computed for each dataset. 3. A meta-dataset is created and a meta-model is learned. [5]

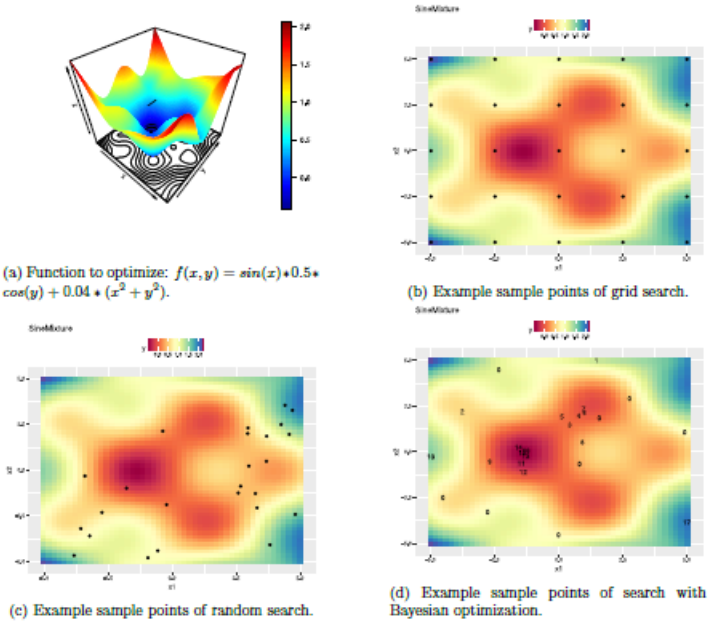
find the right ones and there are many different approaches to tackle preprocessing. All of these are discussed in their own subsection to briefly explain them.

2.3.1 Meta-Learning

Machine learning algorithms show different behaviour for different tasks. Meta-learning tries to find out how their performance changes between those tasks (Figure 2.3.1). It tries to link the algorithm with data sets it would do good for and tries to find which hyperparameters give a good performance. In combination with the machine learning pipelines, meta-learning would try to find the best ones available. Since not only algorithms must be selected, but also hyperparameters must be optimized and preprocessing must be done, the time and space needed for meta-learning explodes. This can be lessened when removing bad pipelines and limiting the range of hyperparameters, machine learning- and preprocessing algorithms as much as possible.

Features of the meta-learning phenomenon are used to predict the performance. There are three types of these meta-features. The first type is simple, statistical and information-theoretic. They can be a basic feature of the data set, as well as a value after a statistical computation or a specific theoretical value. The second meta-feature type can be called landmarks. Landmarks give the performance of algorithms, how well they are doing with the given data set. The last meta-feature category is model-based. Specific characteristics of the used model can be used as meta-features as well. [37, 38]

For using those meta-features in picking the best machine learning algorithm meta-learners can be used. Meta-learners are algorithms that choose between the possible choices. There are four ways of doing that. The first is plainly choosing the best algorithm in the set, this choice speeds up the process but is prone to being a bad choice. Second a subset of good algorithms can be chosen, which is slower, but has a higher chance to give a good outcome. Thirdly the algorithms can be ranked, which makes the chance of picking a good algorithm quicker starting at the top. Fourth is to use estimations of performance which gives information expectations over all algorithms. [39]



(a) Function to optimize: $f(x,y) = sin(x)*0.5*cos(y) + 0.04 * (x^2 + y^2)$.

(b) Example sample points of grid search.

(c) Example sample points of random search.

(d) Example sample points of search with Bayesian optimization.

Figure 2: Examples of the three explained hyperparameter optimization techniques. The numbers in the bayesan optimization show the iteration of sampling. [5]

2.3.2 Hyperparameter Optimization

As discussed before, machine learning algorithms have hyperparameters. These type of parameters are very sensitive and can change the algorithm performance greatly, hence the hyper- prefix. These hyperparameters can be nonlinear and nonconvex which results in it being hard to find the optimal value. They are many different variable types of hyperparameters, which makes standardizing optimization hard. They can also be dependent on each other and therefore have useless combinations. At last they can change the computation time drastically for a minor change in parameter. [40]

There are several approaches to find the right values for the hyperparameters. Three approaches will be discussed. At first there is grid search, that checks all possible combinations for hyperparameters on a predefined interval. This approach does effectively check the complete area the optimal solution can be in, however it also takes much computational time due to the combinatorial explosion principle. [41] A second approach is the random search, when values are chosen for each hyperparameter at random. It is proven that this search is better in an empirical and theoretical way, due changing some hyperparameters hardly making any difference. [42]

A more advanced way of optimizing hyperparameters is Bayesian Optimisation. This hyperparameter optimization approach tries to use earlier results to find the best possible location for the hyperparameters to be optimized. At the start, with random sampling several sample points are measured. An acquisition function, with the input from earlier samples, which combination of hyperparameters should be tested next. It balances between trying to explore areas with good performance and trying to explore bigger areas for possible other good areas. [43] The three optimization techniques are shown together to explain the difference (Figure 2.3.2).

2.3.3 Preprocessing

Preprocessing is one of the three aspects of autoML. Whereas the goal is important when choosing a type of machine learning algorithm, preprocessing algorithms always need to be fine-tuned for the available data set, as every set is different. Data sets, and specially biomedical ones for this

project, have specific challenges that can be solved using preprocessing (subsection 2.1). These challenges can be classified in two regions. The first one is about problems with the data, The second one is data preparation. [44]

From a preprocessing point of view, there can be three different types of problems with data (Figure 3). The first problem is that there is too much data. The data can be noisy, irrelevant, too big, different types of data can be present and feature extraction still has to be done. The data for this project is an example for noisy data (subsection 2.1), as 5% of the data is estimated to be wrong.

A second type is the opposite of having too much data. There can also be too little data. Values of an attribute or complete attributes can be absent from the data. The number of data points can also be very low. For the example data set, there were many cases where attributes were missing.

A third problem with the data can be that it is fractured. Multiple separate data sets can be incompatible, come from multiple sources or are on different processing levels. When taking the bariatric data as an example, it stems from two different data sets. A challenge is to combine these two as one data set, as one, while they are not specifically made for that.

To tackle those three data problems, again three types of techniques can be used. At first data can be transformed to become more usable. The most important transformation is noise removal. It can be removed with smoothing function [45, 46], or a more advanced machine learning technique to also detect it. [47]

A second preprocessing technique is gathering data if needed. Data selection is important, as it could be that not all data is as relevant as all the other data. Important techniques to be mentioned are principal component analysis, that checks the relevance for every feature in the data set. [48]

At last new information can be generated, if needed. This can be done by simulation or by adding new features. Data points can be fused to become a new point. This way more data is available for data analysis. Also when values are missing, several techniques can be used for value imputation. Extrapolation can be done, using regression to estimate its value. Other methods are based on nearest neighborhood frameworks. [49]

Aside from data problems, another reason for preprocessing can be present. [44] Data can also be very raw. STo reach a certain goal, a data set has the necessary iformation, but only indirectly. An example would be hypertension. To know if someone has hypertension, its blood pressure must be measured and checked if that value is high enough. this preprocessing is highly data set specific, as computers do not know the specifications of blood pressure for someone having hypertension.

2.4 Tree-based Pipeline Optimization Tool

A tool that implements autoML is tree-based pipeline optimization tool (TPOT). It uses the machine learning pipelines and evolutionary optimization to find the best solution for every data set. This evolutionary optimization is done by genetic programming. Genetic programming evolves possible solutions to find a better solution. This evolution is done by first evaluating them and selecting the best ones to continue to the next generation. Then both crossovers between and mutations on possible solutions are performed. After that again evaluation and selection, followed by crossovers and mutations, take place a number of times until a certain quality is found, time has run out or another ending condition has been met.

TPOT makes use of this genetic programming with using the machine learning pipelines in a tree (Figure 4). TPOT consists of preprocessing and machine learning algorithms, that form the backbone of the pipelines. Their hyperparameters and the data set are the variables. TPOT makes mostly use of the machine learning and preprocessing algorithms of skikit-learn from Python in which it is also written.

TPOT has three different types of mutations within one pipeline.The first one is insertion, inserting a primitive somewhere in the tree. An example would be the insertion of an additional preprocessing algorithm. The second one is replacement, which replaces a random terminal. It

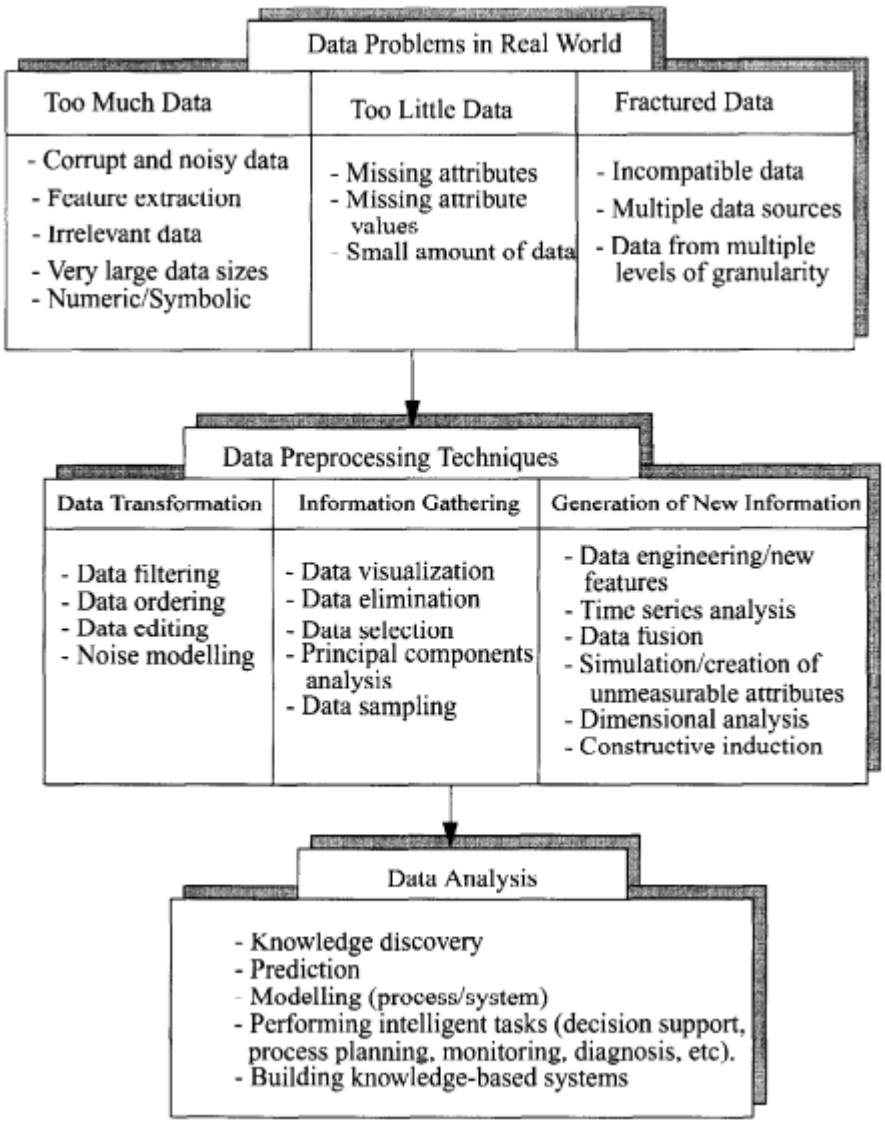


Figure 3: A schema that shows the process of preprocessing. [44]

can for example change a binary hyper parameter from true to false. The third one is shrinking. A primitive is replaced by a terminal. For example a preprocessing step can be replaced by just raw data. This different mutations can all be seen visually (Figure 5)

TPOT also focuses on mutations between two pipelines through the means of crossovers. Between two pipelines, sub-trees and primitives can be changed, given that the both pipelines remain valid (Figure 6). Every time a crossover is performed, two separate pipelines are used and changed, creating two new ones.

Comparing the possibilities from TPOT and the challenges in biomedical data, it seems that TPOT has some implementations to tackle them. It has several different scalers (StandardScaler, RobustScaler, MinMaxScaler) to tackle feature heterogeneity between different data sets and errors. It also has some feature selection operators to tackle errors (VarianceThreshold, SelectKBest, SelectPercentile). For missing values, the algorithm imputes the median as estimation for the missing value.

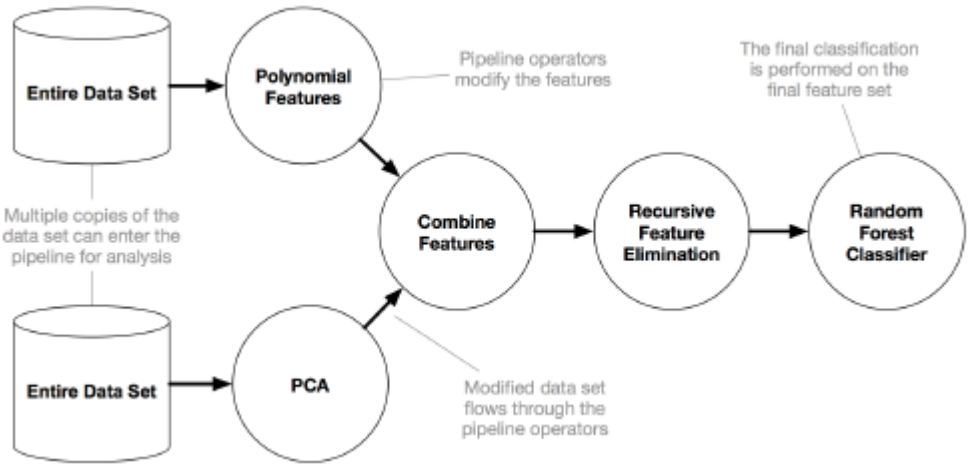


Figure 4: An example of a machine learning pipeline in TPOT. It only shows the primitive algorithms and not hyperparameter terminals. At the root is the machine learning algorithm. [5]

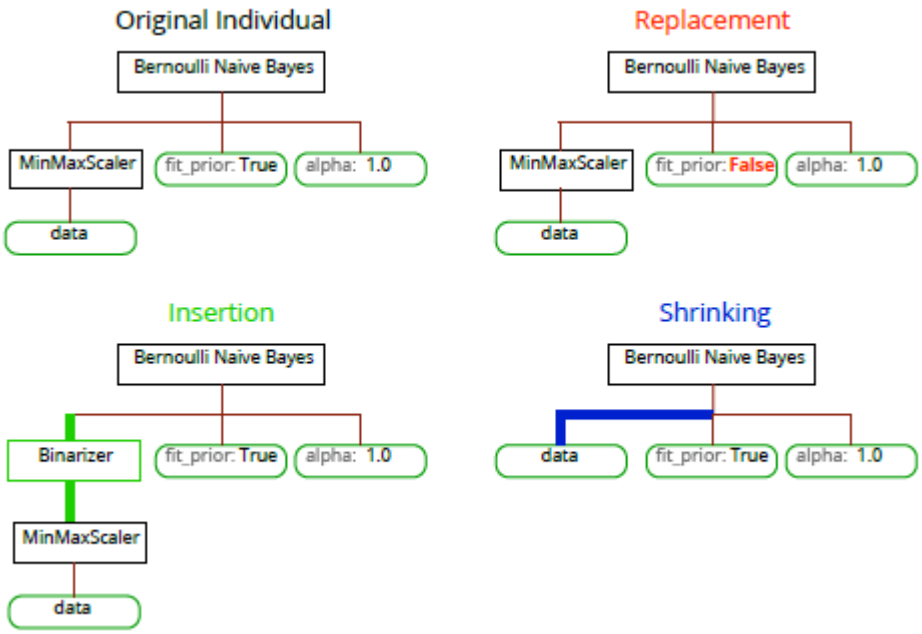


Figure 5: Examples of the three mutations in the TPOT algorithm: insertion, replacement and shrinking. [5]

3 Research Question

TPOT has shown to give promising results for several different data sets. [5] For this project the focus will be on biomedical data sets (subsection 2.1) and how it handles specific problems in these data sets. An example data set (subsection 2.2) is taken for analysis and to create suggestions for future extensions of TPOT. The research question for TPOT will be the following.

How does TPOT perform on specific biomedical data set problems and how can it be improved on them?

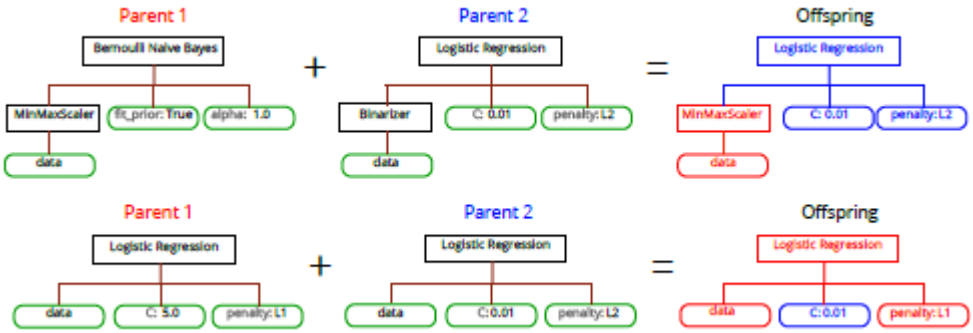


Figure 6: An example of a TPOT crossover [5]

3.1 Hypothesis

Knowing that some algorithms exist to tackle the problems with biomedical data sets, the first step should be to find out whether TPOT can process the data set without any problems. It is expected to succeed in that, as TPOT showed good results for nono-biomedical data sets. The skin disease data set (subsection 2.2) had three biomedical data challenges: data heterogeneity, high dimensionality and data quality. These challenges are discussed separately

With regards to data heterogeneity, TPOT has no ways to use multiple data sets as an input and aligning them. This needs to be done manually by preprocessing. Since the data stems from different experiments, bias should be present. This bias can be tackled by some forms of preprocessing that are available in TPOT. Examples are using normalizers, several types of scalers, thresholds and binarizers. These preprocessors are done only after merging the data sets though, which could hinder the effectivity. This challenge would not be tested on specifically, however could hinder the final scoring of pipelines.

TPOT has ways to tackle high dimensionality with preprocessing. Principal Components Analysis (PCA), Feature Agglomeration, Independent Components Analysis (ICA), Recursive Feature Eliminations (RFE) are four examples of dimensionality reduction. These dimensionality reductions are often used in biomedical data analysis and could prove useful. A benefit of these methods to reduce high dimensionality is the simplicity to extract important genes from the pool after being preprocessed. Other ways TPOT tackles the high dimensionality is with several machine learning techniques as decision trees and random forests. These techniques filter out the best genes for classifications.

The third challenge was about the number of samples. There was a very low number of samples which could cause problems for the machine learning algorithm to correctly find an output. No algorithms were present in TPOT to tackle that, so this could cause issues.

All things considered TPOT should be able to handle several biomedical data set challenges sufficiently, whereas others are probably not present at all. Future development can be suggested to improve TPOT and have it sufficiently tackle those biomedical data problems, as well.

4 Methods

The approach to answer the research question was split in three different parts. At first preprocessing was done to align the nine data sets as much as possible. Secondly the data sets were used as input in the TPOT classifier to find the best possible pipeline and some analysis was done to evaluate that. Third and last the data set was combined as much as possible and run through the model again.

All programming in this project has been done using Python, with the help of anaconda. TPOT was written in python and therefore using Python would seriously reduce compatibility issues. Anaconda was used for for the same reasons, as with anaconda it is fairly simple to import packages used by TPOT and packages that are imported for preprocessing and analysis. Jetbrains Pycharm was used as an environment mainly because of coding assistance and previous experience with it.

For running the experiments only a cpu was available. This meant that memory issues could be present. After some testing on basic data sets (e.g. MNIST), memory gave no issues and was not expected to give issues after using data sets of a similar size.

4.1 Preprocessing

Preprocessing was a big part of the project. Even though all data was generated with the NCBI database [34] in mind, the raw data had to be pruned and all of them in a different way. This pruning consisted of removing irrelevant data, extracting essential data and prepraring it for tpot to use.

To download the data, the NCBI database was used (Table 1). An advantage of the NCBI database is that the data is made available in different file types. The base type is a *.cel* type and all other types were based on that one. Other types were *SOFT*, *MINiML* and also just a regular matrix (*.txt*) file. Since only a matrix with the data and type of skin for each sample was needed, the most basic (*.txt*) file was used. Nine different (*.txt*) files were downloaded from the database and stored for preprocessing

The data was one big matrix with for every row an undetermined number of columns. The rows could be split into four different parts: title, set attributes, sample attributes and experiment values. Each part had their own starting row followed by one specific attribute every row. Therefore at first the data was split into these four sections, so further pruning could be done on every section separately. The title and set attributes only consisted of information how to manually locate relevant data locations. Therefore after reading them, these two parts were discarded for future preprocessing.

The sample details contained several attributes for every sample. This part consisted of much irrelevant data, as well, with the exception of skin type. This skin type therefore needed to be extracted first before discarding this part. This skin type had to be manually extracted out of every sample title, as no attribute for skin type was present. The experiment values were more straightforward. The first row consisted of the sample IDs, that could be linked to the sample IDs of the sample details. Every following row consisted of a specific gen as well as the values of every sample.

Preparation of the data was minor due to efficient data extraction. The experiment values were extracted as a matrix and therefore immediately available for the TPOT classifier. For the skin types, three categories were made and a list was made with those categories, corresponding to the samples and their experiment values. The data sets and available skin types (table 1) were mapped as good as possible on the list:

(0) Normal skin. The normal skin values were extracted from patients not suffering from a skin disease. In case of no mention whether skin was normal or non-lesional, it was expected to be normal

(1) Non-lesional skin. Skin of a skin disease patient not affected by it.

(2) Lesional skin. Skin of a skin disease patient that is affected by it.

(3) Dummy variable. Variable specific for that data set that is not compatible with the others

Not all data sets could efficiently be mapped. The 'GSE78097' set had two types of lesional (mild and severe) and no normal skin. The 'GSE27887' had pre- and post treatment non-lesional and lesional skin. For both of these sets, a manual ordering was made, so these data sets should be treated differently when combined

Table 2: All TPOT Algorithms used for the TPOT classifier function. Not only the algorithms differ, but also the hyperparameters within these algorithms.

Algorithm type	**Specification**	**Algorithms**
Classifier	Naïve Bayes	GaussianNB, BernoulliNB, MultinomialNB
	Decision Tree	DecisionTree, ExtraTrees, RandomForest, GradientBoosting
	Nearest Neighbor	KNeighbors
	Support Vector Machines	LinearSVC
	Logistic Regression	Logistic Regression
Preprocessors	Scaler	Binarizer, MaxAbsScaler, MinMaxScaler, Normalizer, RobustScaler, StandardScaler
	Feature reduction	PCA, FastICA, RBFSampler, Nystroem, FeatureAgglomeration
	Feature Modifier	Polynomial, OneHotEncoder,ZeroCount
	Feature Selectors	SelectFwe, SelectPercentile, VarianceThreshold, RFE, SelectFromModel

4.2 TPOT

After the data sets had been preprocessed, they were available for TPOT calculations. The experimental values and skin types should be used as input for the TPOT classifier whereas the values are the input features and the skin types the output. Before actually using the data, first it should be split into a training and test set. Even though TPOT itself also uses a training and test set to prevent overfitting of data, a small test set still was used to calculate the score on data TPOT had not seen itself, yet. Since the number of samples was very low to begin with, only 90% of the data was regarded as training data and only 10% as test data.

Since classification was needed, the TPOT classifier was used which made use of several algorithms. These algorithms were mostly borrowed from sklearn and some were created by TPOT itself (Table 2). Before using TPOT, several parameters had to be selected. The selected parameters all revolve around the genetic algorithm used for selecting the best machine learning pipeline (subsection 2.3):

- *generations*, the number of generations the genetic algorithm should run before terminating. If the number of generations had been reach the TPOT classifier ended. the number of generations was set on 100.

- *population_size*, the number of pipelines that needed to be stored for further generations. After every generation the worst pipelines were discarded. The number of modifications was also linked to the number of modifications to the pipelines, so every generations 10 modifications were made. The population size was set on 10.

- *max_time_mins*, the time limit the TPOT classifier is allowed to find the best pipeline. If it still runs when the time limit has been reached it will end pre-emptively. Due to severe time constraints, the time limit was set on 120 minutes.

After the best pipelines had been chosen, they were stored in a separate file. These files consisted of both the chosen algorithms and their hyperparameters as well as the score after fitting the test size on it. Since TPOT only chose the best possible pipeline, if it would be used it should be fitted another time for the data set for further calculations.

To analyse the created pipelines, every data set was fitted on every pipeline. Again a training and test set was created to find out how well the pipelines would behave for very data set. The acquired score could be used to show how well that pipeline would behave for every data set. Every data set was fitted 10 times and the acquired scores were averaged to better represent the actually quality of the pipeline.

4.3 Data Set Combination

At last all possible data sets were combined to create one big data set with more samples. Since not all of them were applicable to add, only the data sets with the names 'GSE13355', 'GSE30999', 'GSE34248', 'GSE41662' and 'GSE14905' were chosen. Psoriasis data sets were chosen instead of atopic dermatitis simply because more of those were available. After combining these data sets the final number of samples was 508 with the following distribution:

- *Normal skin:* 85 samples
- *Non-lesional skin:* 209 samples
- *Lesional skin:* 214 samples

To run this with TPOT classifiers new input parameters should be chosen. Since this data set was about three times larger than the largest data set run so far and the differences between data sets had to be overcome as well, a larger running time had to be given for this case. The following parameters were given:

- *generations:* 100
- *population_size:* 40
- *max_time_mins:* 480

After the TPOT classification had been run, it was analysed for the data set. This was done by fitting it ten times and averaging the acquired scores. The eight original pipelines were all being fitted for the data set the same way, to find out if they still produced good results with more data points from different data sets.

5 Results

The results were split into two parts. Since preprocessing did not have any results to be presented (the preprocessed dataset itself was the result), only TPOT and combination of the data sets were presented.

5.1 TPOT

Before acquiring the results there were several problems with using the data set in TPOT. At first several TPOT algorithms had trouble with using several algorithms on the high dimensional data set. Some required almost all memory making the TPOT classifier much slower than needed. The preprocessing algorithm *FeatureAgglomeration* even froze the computer, requiring it to restart. Therefore this algorithm was manually removed from all possibilities in the TPOT classifier. The data set 'GSE36842' also had problems finding an optimal pipeline because of unknown reasons and therefore was also removed from the pipeline selection.

The results for the eight remaining data sets were different from each other (Table 3). In all cases the running time was the biggest constraint, so new runs should be done in longer time. The most picked classifier seemed to be *RandomForestClassifier*, being picked three times to find possible better results. The hyperparameters did not seem very similar for these three though. *LinearSVC* was also picked twice, with different hyperparameters. Interestingly enough a preprocessing algorithm was used only once, which was the *Binarizer* algorithm. This preprocessing alogrithm was not a feature reduction algorithm, while those algorithms were more expected to be present in the final pipelines. Looking at final scores of these pipelines, most of them were close to a perfect score, two of them even being a perfect classification score. Data sets 'GSE13355' and 'GSE32924' settled for a far lower score around 0.83 and for data set 'GSE27887'a score of only 0.51 was the best the TPOT classifier could find.

Table 3: All discovered pipelines with the TPOT classifiers for the mentioned data sets as well as the scores for those pipelines. The sample size is given for comparison, too.

Data set	Sample size	Score	Pipeline
GSE13355	180	0.8333793341383474	Binarizer(threshold=0.7) RandomForestClassifier(bootstrap=False, criterion="entropy", max_features=0.35, min_samples_leaf=3, min_samples_split=18, n_estimators=100))
GSE30999	170	0.9804301075268818	KNeighborsClassifier(n_neighbors=9, p=2, weights="distance")
GSE34248	28	0.9666666666666668	RandomForestClassifier(bootstrap=True, criterion="entropy", max_features=0.9, min_samples_leaf=3, min_samples_split=6, n_estimators=100)
GSE41662	48	1.0	LinearSVC(C=0.001, dual=True, loss="hinge", penalty="l2", tol=0.01)
GSE78097	33	1.0	RandomForestClassifier(bootstrap=False, criterion="gini", max_features=1.0, min_samples_leaf=4, min_samples_split=10, n_estimators=100)
GSE14905	82	0.9733333333333334	LinearSVC(C=5.0, dual=True, loss="squared_hinge", penalty="l2", tol=0.001)
GSE32924	33	0.8283333333333334	GradientBoostingClassifier(learning_rate=1.0, max_depth=8, max_features=0.75, min_samples_leaf=4, min_samples_split=3, n_estimators=100, subsample=0.7)
GSE27887	35	0.5142857142857142	DecisionTreeClassifier(criterion="gini", max_depth=4, min_samples_leaf=1, min_samples_split=10)

Since the data sets were similar. The data sets were tested on each other multiple times to find out which pipeline worked best on which data set (Figure 7). Those scores showed that no pipeline is best for any data set. Furthermore, they showed that even after trying different pipelines, they show very similar behaviour. For almost every data set no clear best or for example top three could be chosen.

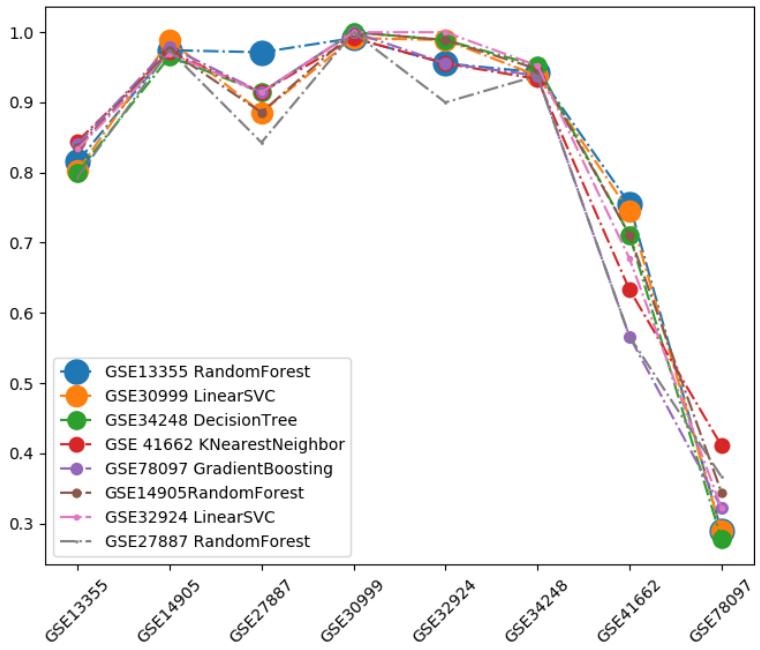


Figure 7: The scores of the pipelines after fitting them on every data set.

5.2 Data Set Combination

Also when combining the data set combinations, TPOT stopped trying to find the best pipeline after the time limit has been reached. If a factor of 10^2 would be added to the time (so 48000 minutes), all generations should be executed properly. This would result in 33 days of computation, though, which was not available. The final pipeline was the following:

> *Best pipeline:* LogisticRegression(C=0.0001, dual=True, penalty='l2'), GradientBoosting-Classifier(learning_rate=0.1, max_depth=5, max_features=0.05, min_samples_leaf=12, min_samples_split=19, n_estimators=100, subsample=0.5)

The best pipeline being a combination of logistic regression and gradient boosting classifier seems like a plausible best pipeline. Both are often used for high dimensional data. Also, when looking at the hyperparameters of the gradient boosting classifier, the decision trees made for it can become quite big (a minimum of 5 depth and 12 leaf nodes). The logistic regression could make choices for the decision tree easier.

The score for the newly created pipeline is 0.913725490196. This score is not as high as the score for some data sets separately, which can be explained by the bias entered when combining them. The score has a high success rate, though, so it definitely is not bad. The pipelines created for the original data sets were also checked for the combined set to find a difference (Table 4). These show that the final created data set did show better results, although several algorithms came close.

6 Conclusion

To give an answer to the research question (section 3) how TPOT would perform with typical biomedical data set problems, a two sided answer should be given. It is able to cope with some data set problems, however also shows issues with them.

Table 4: The scores when using the pipelines from the separate data sets with the combined set. The score for their own data set set are added as well as the algorithms used.

Data set	Algorithms	Own score	Combined Data set score
GSE13355	Binarizer RandomForest	0.8333793341383474	0.674509803922
GSE30999	KNearestNeighbor	0.9804301075268818	0.860784313725
GSE34248	RandomForest	0.9666666666666668	0.911764705882
GSE41662	LinearSVC	1.0	0.901960784314
GSE78097	RandomForest	1.0	0.858823529412
GSE14905	LinearSVC	0.9733333333333334	0.905882352941
GSE32924	GradientBoosting	0.8283333333333334	0.835294117647
GSE27887	DecisionTree	0.5142857142857142	0.882352941176

A first issue came up when running the high dimensional data. TPOT had memory issues when using several algorithms, even concluding in one algorithm being excluded from the possible outcomes. With time being a constraint for finding the best possible pipeline, it is not desired that several options take much longer to compute.

Where on the one hand, the high dimensionality was an issue, the low number of samples seemed an issue, as well. Test sets could accidentally satisfy the fitted pipeline much easier when only a small subset was used. No really possibilities are available to tackle that. An example would be to create more samples using the known ones.

TPOT found good pipelines despite the issues. Five of the 8 data sets in the end had a pipeline with a scoring of (close to) 1. This means that it did succeed finding good pipelines for more than half of the data sets. This shows that there definitely is potential for TPOT to be used for biomedical data sets.

TPOT also found a good pipeline for the combined data set. The score was reasonable and that pipeline outperformed most of the pipelines generated for the separate data sets.

All things considered TPOT did perform reasonably well for these biomedical data sets. It showed results that are definitely usable for further research. When some more changes would be added to address memory issues, adding the options of choosing possible algorithms and adding the possibility of creating more samples, TPOT become a very good tool for biomedical engineers to use.

7 Discussion

The biomedical world strongly benefits from using the advanced techniques created by the data mining world. These techniques usually enter the biomedical data analysis projects later, due to a gap between them. This can be achieved by creating better communication between the worlds, so the data miners understand and implement important aspects of the biomedical research and the biomedical engineers do the same with data mining techniques. Projects like this one help filling that gap. Continuing this more breakthroughs can be achieved for better understanding and treatments in the hospital.

During this project time was a severe constraint. At first different data was supposed to be made available. This data was bariatric data from the Catharina Hospital in Eindhoven, which focussed on different biomedical data challenges. [50] When it became clear that it was not possible to have this data in time, new data had to be searched. This meant that the implementation phase of the project did start in January instead of November. On top of that, the preprocessing took up quite some time as did solving the memory issues and finding out which algorithm froze the cpu. Because of that the experiments were run much shorter than initially planned to run. This worsened the results as TPOT is not expected to successfully find an optimization. A future work on this project extending it for a period of time would be of great benefit.

As mentioned in the conclusion (Section 6), TPOT would benefit from extensions regarding high dimensionality, low data volume and high data heterogeneity. This extension would be a good topic for further research to find out what types of extensions would help in those cases. These extensions could be adding new algorithms or new algorithm features, but also adding limitations to the search space and research warm start possibilities (starting with predefined pipelines).

Only a selection of biomedical data set issues were present for this skin disease data set. Issues such as data complexity, volume or different aspects of quality could be addressed with a different data set. A clinical data set with blood markers and manual subscriptions would be an example of a data set with quality challenges, whereas a data set with MRI images would be interesting for testing complexity. Future projects could be based around those.

A last not discussed characteristic is showing the links between input with output. Creating a model that correctly links input and output is nice, however in the biomedical world there is more to it than only creating this model. For the skin disease data set it would be very helpful if the conclusion would have given several genes that were important for diagnosing psoriasis or atopic dermatitis. Those genes can be further tested in what way they stimulate the expression of those diseases to gain more insights to possibly find a remedies for that. Whereas some classification techniques clearly show which features were used (decision trees, random forests), others create a swamp in which it is much harder to find them (support vector machines). A possible extension for TPOT to address this characteristic would be very beneficial.

References

[1] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum, and F. Hutter, "Efficient and robust automated machine learning," in *Advances in Neural Information Processing Systems*, pp. 2962–2970, 2015.

[2] R. S. Olson and J. H. Moore, "Tpot: A tree-based pipeline optimization tool for automating machine learning," in *Workshop on Automatic Machine Learning*, pp. 66–74, 2016.

[3] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic programming: an introduction*, vol. 1. Morgan Kaufmann San Francisco, 1998.

[4] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a tree-based pipeline optimization tool for automating data science," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, pp. 485–492, ACM, 2016.

[5] P. Gijsbers, "Automatic construction of machine learning pipelines," 2017.

[6] J. D. Bronzino and D. R. Peterson, *Biomedical engineering fundamentals*. CRC press, 2014.

[7] M. Bramer, *Principles of data mining*, vol. 180. Springer, 2007.

[8] H. Chen, S. S. Fuller, C. Friedman, and W. Hersh, *Medical informatics: knowledge management and data mining in biomedicine*, vol. 8. Springer Science & Business Media, 2006.

[9] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: review, opportunities and challenges," *Briefings in Bioinformatics*, p. bbx044, 2017.

[10] D. Blythe, "Rise of the graphics processor," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 761–778, 2008.

[11] C. Turkay, F. Jeanquartier, A. Holzinger, and H. Hauser, *On Computationally-Enhanced Visual Analysis of Heterogeneous Data and Its Application in Biomedical Informatics*, pp. 117–140. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[12] A. Holzinger and I. Jurisica, *Knowledge Discovery and Data Mining in Biomedical Informatics: The Future Is in Integrative, Interactive Machine Learning Solutions*, pp. 1–18. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[13] W. Dubitzky, M. Granzow, and D. P. Berrar, *Fundamentals of data mining in genomics and proteomics*. Springer Science & Business Media, 2007.

[14] Y. Peng, Z. Wu, and J. Jiang, "A novel feature selection approach for biomedical data classification," *Journal of Biomedical Informatics*, vol. 43, no. 1, pp. 15 – 23, 2010.

[15] I. Yoo, P. Alafaireet, M. Marinov, K. Pena-Hernandez, R. Gopidi, J.-F. Chang, and L. Hua, "Data mining in healthcare and biomedicine: A survey of the literature," *Journal of Medical Systems*, vol. 36, pp. 2431–2448, Aug 2012.

[16] R. Bellazzi, M. Diomidous, I. N. Sarkar, K. Takabayashi, A. Ziegler, A. T. McCray, *et al.*, "Data analysis and data mining: current issues in biomedical informatics," *Methods of information in medicine*, vol. 50, no. 6, p. 536, 2011.

[17] D. Otasek, C. Pastrello, A. Holzinger, and I. Jurisica, *Visual Data Mining: Effective Exploration of the Biological Universe*, pp. 19–33. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[18] K. J. Cios and G. W. Moore, "Uniqueness of medical data mining," *Artificial Intelligence in Medicine*, vol. 26, no. 1, pp. 1 – 24, 2002. Medical Data Mining and Knowledge Discovery.

[19] L. Marenco, T.-Y. Wang, G. Shepherd, P. L. Miller, and P. Nadkarni, "Qis: A framework for biomedical database federation," *Journal of the American Medical Informatics Association*, vol. 11, no. 6, pp. 523–534, 2004.

[20] V. Y. Bichutskiy, R. Colman, R. K. Brachmann, and R. H. Lathrop, "Heterogeneous biomedical database integration using a hybrid strategy: a p53 cantcer research database," *Cancer informatics*, vol. 2, p. 277, 2006.

[21] W. Sperzel, R. Abarbanel, S. Nelson, M. Erlbaum, D. Sherertz, M. Tuttle, N. Olson, and L. Fuller, "Biomedical database inter-connectivity: an experiment linking mim, genbank, and meta-1 via medline.," in *Proceedings of the Annual Symposium on Computer Application in Medical Care*, p. 190, American Medical Informatics Association, 1991.

[22] F. Aubry, S. Badaoui, H. Kaplan, and R. D. Paola, "Design and implementation of a biomedical image database (bdim)," *Medical Informatics*, vol. 13, no. 4, pp. 241–248, 1988.

[23] D. Windridge and M. Bober, *A Kernel-Based Framework for Medical Big-Data Analytics*, pp. 197–208. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[24] S. R. Rapp, S. R. Feldman, M. L. Exum, A. B. Fleischer, and D. M. Reboussin, "Psoriasis causes as much disability as other major medical diseases," *Journal of the American Academy of Dermatology*, vol. 41, no. 3, pp. 401–407, 1999.

[25] S. Jowett and T. Ryan, "Skin disease and handicap: an analysis of the impact of skin conditions," *Social science & medicine*, vol. 20, no. 4, pp. 425–429, 1985.

[26] R. P. Nair, K. C. Duffin, C. Helms, J. Ding, P. E. Stuart, D. Goldgar, J. E. Gudjonsson, Y. Li, T. Tejasvi, B.-J. Feng, *et al.*, "Genome-wide scan reveals association of psoriasis with il-23 and nf-κb pathways," *Nature genetics*, vol. 41, no. 2, pp. 199–204, 2009.

[27] M. Suárez-Farinas, K. Li, J. Fuentes-Duculan, K. Hayden, C. Brodmerkel, and J. G. Krueger, "Expanding the psoriasis disease profile: interrogation of the skin and serum of patients with moderate-to-severe psoriasis," *Journal of Investigative Dermatology*, vol. 132, no. 11, pp. 2552–2564, 2012.

[28] J. Bigler, H. A. Rand, K. Kerkof, M. Timour, and C. B. Russell, "Cross-study homogeneity of psoriasis gene expression in skin across a large expression range," *PLoS One*, vol. 8, no. 1, p. e52242, 2013.

[29] J. Kim, R. Bissonnette, J. Lee, J. C. da Rosa, M. Suárez-Fariñas, M. A. Lowes, and J. G. Krueger, "The spectrum of mild to severe psoriasis vulgaris is defined by a common activation of il-17 pathway genes, but with key differences in immune regulatory genes," *Journal of Investigative Dermatology*, vol. 136, no. 11, pp. 2173–2182, 2016.

[30] Y. Yao, L. Richman, C. Morehouse, M. De Los Reyes, B. W. Higgs, A. Boutrin, B. White, A. Coyle, J. Krueger, P. A. Kiener, *et al.*, "Type i interferon: potential therapeutic target for psoriasis?," *PloS one*, vol. 3, no. 7, p. e2737, 2008.

[31] M. Suárez-Fariñas, S. J. Tintle, A. Shemer, A. Chiricozzi, K. Nograles, I. Cardinale, S. Duan, A. M. Bowcock, J. G. Krueger, and E. Guttman-Yassky, "Nonlesional atopic dermatitis skin is characterized by broad terminal differentiation defects and variable immune abnormalities," *Journal of Allergy and Clinical Immunology*, vol. 127, no. 4, pp. 954–964, 2011.

[32] S. Tintle, A. Shemer, M. Suárez-Fariñas, H. Fujita, P. Gilleaudeau, M. Sullivan-Whalen, L. Johnson-Huang, A. Chiricozzi, I. Cardinale, S. Duan, *et al.*, "Reversal of atopic dermatitis with narrow-band uvb phototherapy and biomarkers for therapeutic response," *Journal of Allergy and Clinical Immunology*, vol. 128, no. 3, pp. 583–593, 2011.

[33] J. K. Gittler, A. Shemer, M. Suárez-Fariñas, J. Fuentes-Duculan, K. J. Gulewicz, C. Q. Wang, H. Mitsui, I. Cardinale, C. de Guzman Strong, J. G. Krueger, *et al.*, "Progressive activation of t h 2/t h 22 cytokines and selective epidermal proteins characterizes acute and chronic atopic dermatitis," *Journal of Allergy and Clinical Immunology*, vol. 130, no. 6, pp. 1344–1354, 2012.

[34] R. Edgar, M. Domrachev, and A. E. Lash, "Gene expression omnibus: Ncbi gene expression and hybridization array data repository," *Nucleic acids research*, vol. 30, no. 1, pp. 207–210, 2002.

[35] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: Combined selection and hyperparameter optimization of classification algorithms," in *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 847–855, ACM, 2013.

[36] L. Kotthoff, C. Thornton, H. H. Hoos, F. Hutter, and K. Leyton-Brown, "Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka," *Journal of Machine Learning Research*, vol. 17, pp. 1–5, 2016.

[37] P. Brazdil, J. Gama, and B. Henery, "Characterizing the applicability of classification algorithms using meta-level learning," in *European conference on machine learning*, pp. 83–102, Springer, 1994.

[38] R. Vilalta, C. G. Giraud-Carrier, P. Brazdil, and C. Soares, "Using meta-learning to support data mining.," *IJCSA*, vol. 1, no. 1, pp. 31–45, 2004.

[39] P. Brazdil, C. Giraud-Carrier, C. Soares, and R. Vilalta, "Development of metalearning systems for algorithm recommendation," *Metalearning: Applications to Data Mining*, pp. 31–59, 2009.

[40] M. Claesen and B. De Moor, "Hyperparameter search in machine learning," *arXiv preprint arXiv:1502.02127*, 2015.

[41] C.-W. Hsu, C.-C. Chang, C.-J. Lin, *et al.*, "A practical guide to support vector classification," 2003.

[42] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[43] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, pp. 2951–2959, 2012.

[44] A. Famili, W.-M. Shen, R. Weber, and E. Simoudis, "Data preprocessing and intelligent data analysis," *Intelligent data analysis*, vol. 1, no. 1-4, pp. 3–23, 1997.

[45] R. Somorjai, M. Alexander, R. Baumgartner, S. Booth, C. Bowman, A. Demko, B. Dolenko, M. Mandelzweig, A. Nikulin, N. Pizzi, *et al.*, "A data-driven, flexible machine learning strategy for the classification of biomedical data," *Artificial intelligence methods and tools for systems biology*, pp. 67–85, 2004.

[46] A. Karagiannis and P. Constantinou, "Noise-assisted data processing with empirical mode decomposition in biomedical signals," *IEEE Transactions on Information Technology in Biomedicine*, vol. 15, no. 1, pp. 11–18, 2011.

[47] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise detection and elimination in data preprocessing: experiments in medical domains," *Applied Artificial Intelligence*, vol. 14, no. 2, pp. 205–223, 2000.

[48] Z. Duszak and W. Koczkodaj, "Using principal component transformation in machine learning," in *Proceedings of International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden Germany*, pp. 125–129, 1994.

[49] X. Zhu, S. Zhang, Z. Jin, Z. Zhang, and Z. Xu, "Missing value estimation for mixed-attribute data sets," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 1, pp. 110–121, 2011.

[50] R. Deneer, "Scoring co-morbidity severity in bariatric patients based on biomarkers: a data mining approach," 2017.