Iterative Neural Scoring of Validated Insight Candidates

Allmin Susaiyah[1], **Aki Härmä**[2], **Ehud Reiter**[3] **and Milan Petković**[1]
[1] Eindhoven University of Technology, Netherlands
[2] Philips Research, Eindhoven, Netherlands
[3] University of Aberdeen, Scotland

Abstract

Automatic generation of personalised behavioural insight messages is useful in many applications, for example, health self-management services based on a wearable and an app. Insights should be statistically valid, but also interesting and actionable for the user of the service. In this paper, we propose a novel neural network approach for joint modeling of these elements of the relevancy, that is, statistical validity and user preference, using synthetic and real test data sets. We also demonstrate in an online learning scenario that the system can automatically adapt to the changing preferences of the user while preserving the statistical validity of the mined insights.

1 Introduction

Recently, many health and fitness apps have stormed the market claiming to be able to improve user behaviour by playing the role of an artificial health or fitness agent (Hingle and Patrick, 2016; Higgins, 2016). While the customer base for these apps is in billions, it is still a question if they are effective in doing what they claim. One goal of these applications is to help the user understand the own behaviour by giving actionable insights and advises. In this work we focus on comparative insights that can be considered as categorical statements about a measure in two contexts, for example, stating that a measure X is larger in context A than in context B, see Härmä and Helaoui (2016).

For this, the task of determining if two samples are statistically significantly different is frequently performed. While parametric and non-parametric significance tests have been widely used for such tasks, it remains a challenge to include them into a neural learning pipeline that is both scalable and user-centric. A neural network can act as a universal function approximator and can transfer knowledge from one domain to another. In this work, we consider three domains, namely, statistical significance domain, *interestingness* domain and validity domain. The statistical significance domain includes a non-parametric significance test, namely, the Kolmogorov-Smirnov (KS) test. The interestingness domain that incorporates how a user is interested in knowing about a particular comparative insight. The third domain is the validity of the content for the target application. The system should not produce insights or advises that are harmful to the healthcare goals of the service. This can be best guaranteed by a system where all texts are selected from a pre-generated and manually curated collection of *validated insight candidates*, similarly to the PSVI method introduced in Härmä and Helaoui (2016).

In this work, we train a self-supervised neural network that can be a scalable alternative to traditional non-parametric tests (with 92% accuracy at 5% alpha) and we also show how it can be used to learn user preference on top of statistical significance using an online learning strategy. As these characteristics are essential for highly scalable behavior insight mining (BIM) that finds application is fitness coaching, office behaviour (O'Malley et al., 2012), behaviour change support systems (Braun et al., 2018; Sripada and Gao, 2007), and business insight mining systems (Härmä and Helaoui, 2016), the proposed work is highly relevant.

2 Background

2.1 Desirable Characteristics of Insights

Based on recent literature, an insight should have the several, characteristics, namely, statistical significance (Agrawal and Shafer, 1996; Härmä and Helaoui, 2016), interestingness or personal preferences (Freitas, 1999; Fayyad et al., 1996; Su-

Comparison	Example
time-specific	On **Weekdays** you walk less than on **Weekends**
parameter-specific	Your **heart rate** is higher on Mondays than other days
event-specific	**when you bike**, you spend less calories per minute than when you **run**

Table 1: Examples of comparative insights in BIM

darsanam et al., 2019; op den Akker et al., 2015; Härmä and Helaoui, 2016), Causal confidence (Sudarsanam et al., 2019), surprisingness (Freitas, 1999), actionability or usefulness (Freitas, 1999; Fayyad et al., 1996), syntactic constrains (Agrawal and Shafer, 1996), presentatability (op den Akker et al., 2015) timely delivery (op den Akker et al., 2015), and understandability (Fayyad et al., 1996). Among all of these characteristics the most common ones are statistical validity and interestingness.

2.2 Types of Insights

1. Generic insight: These are insights that talk about a rather common or scientific phenomenon. These are not grounded on the user's behaviour. For example: Excessive caffeine consumption can lead to interrupted sleep as can ingesting caffeine too late in the day.

2. Personalised (Manual/Automated) insight (Reiter et al., 2003): These are insights that are tailored to the user either by a human-in-loop or by an algorithm.

    - Absolute insights: These insights talk about user behaviour in one context. We do not focus on such insights in this paper as they are less actionable.
    - Comparative insights: These insights compare the user behaviour between two contexts (Härmä and Helaoui, 2016) as shown in Table 1.

2.3 Insight Generation Mechanisms

Thousands of insights can be generated from even a simple database by slicing and dicing the data into different views. For example, to generate the insight "On Weekdays you sleep less than on Weekends", the database should have logs of user's sleep duration and corresponding dates. The rows of the database corresponding to weekdays are considered as bin A and those corresponding to weekends are considered as bin B. Relevant filters are used to extract these rows. On comparing the average user's sleep duration in each bin, we find that bin A has a lower value than bin B. Subsequently, a statistical significance test is performed to prove its statistical validity. Similarly, many comparisons could be made between two periods such as:

- Mondays and other days
- Workdays and holidays
- February and March

A detailed description of how insights are generated is explained in Härmä and Helaoui (2016).

2.4 Non Parametric Statistical Significance Tests

The data extracted from the two periods mentioned above come from two non-parametric sample distributions. The two most commonly adapted techniques to determine the statistical significance of such distributions are KS test and Mann-Whitney U (MW)test. The former is based on the shape of the distributions and the latter is based on the ranks of the samples. In this paper we choose the KS test arbitrarily. However, the MW test can also be used instead of that.

2.5 Neural Statistics

Neural networks have been used for wide range applications in Machine Learning such as signal de-noising, image classification, stock prediction, and optical character recognition. The ability of the neural network to learn basically any complex function makes a *universal function approximator*. The simplicity in the way by which a neural network generates an inference makes it a suitable choice for many applications. Additionally, the transfer learning capability of the network (Tao and Fang, 2020; Long et al., 2015; Mikolov et al., 2013) allows us to transfer the pre-learned knowledge of the network to solve different and more complex problems. This inspired us to use the neural network to approximate the statistical significance test.

2.6 Online Learning of User Preference

By permuting different contexts one may often find a large number of statistically significant insights but not all of these insights are useful to the user. Hence the user's preference must be considered before presenting the insights to them. The personal preferences of end-users change with time. Filtering the insights based on statistical validity alone is not sufficient to satisfy their interests. A method to learn a user's preference in a convenient and flexible manner will solve this problem. Online learning technology can train models in a flexible manner while still being deployed in product (Settles, 2009, 2011). There is no existing literature on online learning of user preference nor the learning of statistical validity. Such learning will be of great use in BIM applications.

In this work, we present an online learning strategy that learns user preference while simultaneously maintaining the ability to realise the statistical significance. In our technique, we assume that the user is interested only in one type of insight at any point in time. However, in reality, the user might be interested in multiple types of insights simultaneously. We set this limitation for the sake of simplicity and demonstration only, and by no means is it a limitation of our method.

3 Methodology

The entire methodology was performed in two stages, namely, the self-supervised learning stage and the online learning stage. Although each stage has a different data source, model architecture, training, and validation strategy, they share an important connection. The second stage model is transfer learned from the first. In this section, we describe the above-mentioned stages in detail.

3.1 Self-Supervised Learning Stage

As a first stage, we conceptualised and developed a neural network model that learned rich feature representations to determine the statistical validity of comparative insights. We achieved this by training the model with highly diverse synthetic data. The data generation and model training are described below.

3.1.1 Problem Formulation

Let us consider an insight i that compares two distributions d_1 and d_2. The KS significance test can be represented as a function $f(d1, d2)$ that determines the p-value of d_1 and d_2. If the p-value is less than the significance level α, then, d_1 and d_2 are considered significantly different. We formulated a neural network N that approximates f as shown in Equation 1.

$$f \sim N \tag{1}$$

The neural network learns the function f by minimising the mean squared error loss function J_1 as shown in Eq 2.

$$J_1(\theta) = \frac{1}{n}\sum_{i=1}^{n}(f(d_{1i}, d_{2i}) - N_\theta(d_{1i}, d_{2i}))^2 \tag{2}$$

3.1.2 Data Generation for Base Model Selection

A data-set containing 300000 pairs of histograms of uniform distributions was generated using the NumPy-python package. The number of samples, mean and range of each distribution was chosen randomly. The ground truth labels for each pair of distribution were generated using the p-values of the two-sample KS test. The SciPy-python package was used for this. We compared it with our less optimised implementation of of KS test and found it to give the same p-values. The data-set was subdivided into three equal parts, each for training, validation, and testing. We also made sure that each portion had balanced cases of significant and insignificant pairs.

3.1.3 Finalisation of Base Model Architecture

A domain-induced restriction of comparative insights is that the number of inputs is two and the number of outputs is one. Here, each input is the histogram of distribution and the output is the statistical significance. Based on previous works on similar input/output constraints (Neculoiu et al., 2016; Berlemont et al., 2015), we came up with three neural network architectures, namely, a recurrent neural network (RNNA), a modified RNN (RNNB) and a siamese network (SIAM). The schematics of the RNNA architecture are shown in Figure 1. The layers Ip1 and Ip2 are input layers, each having a fixed size of 100 elements. The layers F1 and F2, are fully connected layers, each with 50 neurons activated by a Leaky Rectified Linear Unit (ReLU) function. In fact, all layers in the network except the Final layer are activated by the Leaky ReLu function. Another level of Fully connected layers, namely, F3 and F4 follow F1 and F2 respectively. We chose the number of neurons in each of these

layers to be 20, which is lesser than the preceding layer, to have a compressed representation of the input signal. This type of compression is believed to help in transforming the input from the spacial domain to the feature domain. The layers F1 and F2 are concatenated and fed to a Simple Bidirectional Recurrent Neural Network (RNN) with 100 units. The rationale behind using an RNN is that the input needs to be considered a sequence rather than a vector as the inputs belong to two different contexts. We added another fully connected layer (F5) having 100 neurons to the output of the RNN. We believe that this layer generates rich features learned from the input data. The final layer is also a fully connected layer with one neuron activated by a thresholded ReLU activation function.

The RNNB model has every layer similar to the RNNA layer, except that it has 100 neurons in the F1 and F2 layers instead of 50. This is to see if increasing neurons would increase performance for a fixed purpose and input size. The SIAM network is also similar to the RNNA architecture, except that the F3 and F4 layers are subtracted rather than being concatenated and the RNN layer is replaced by a fully connected layer with 100 neurons.

3.1.4 Base Model Training and Testing

We trained and validated the three models in a self-supervised manner using the pairs of uniform distributions (histogram). The histogram was squeezed to 100 bins and the minimum and maximum range of histograms are fixed to be the minimum and maximum range of the dataset. This allows all the histograms to be comparable. Uniform distributions were chosen due to their close resemblance to real data that is commonly encountered in insight mining tasks. In total, each of the training, validation and testing phases consisted of 100000 data samples. The training was governed by Adam optimiser with a mean-squared-error loss function. The model that gave the best performance on the test set was considered as the base model. However, in real life, the data could also arise from complex or mixed distributions. Hence we proceeded further with another level of fine training.

3.1.5 Improving the Base Model

To enhance the base model we trained it with more diverse pairs of distributions (histogram) such as Gamma, Gumbel, Laplace, Normal, Uniform and Wald. On the whole, a total of 360000 pairs of distributions were generated and were equally split into training, validation and testing sets. Each of these sets consists of 120000 pairs of distributions (20000 pairs of each distribution). Both inputs of the network are always fed the same type of distribution, but with different parameters. For example, if one input of the network is a normal distribution, the other input is also a normal distribution but with different mean, range, and cardinality. The training labels are generated earlier. The training was governed by Adam optimiser with a mean squared error loss function. Once trained, the model can be used as a smart alternative to statistical significance testing to filter significant insights among all insights.

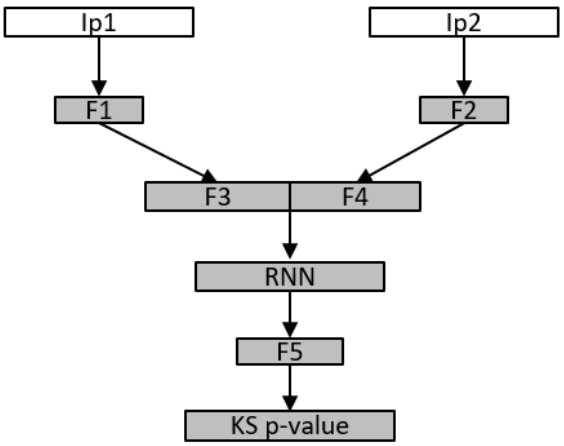

Figure 1: self-supervised neural network architecture for significance testing

3.2 Online Learning Stage

In this stage, we transformed the base model to detect interesting insights while preserving its ability to detect significant insights.

3.2.1 Problem Formulation

In this stage, apart from two distributions d_1 and d_2, we are also interested in the user model ϕ. The user's preference can be represented by a function $p_u(k)$ that generates an interestingness value for a given insight k. This function can also be considered as a user interestingness/preference model. We formulated a transfer learning approach that uses a portion of network N i.e, N' and augments it with features representations generated from another neural network Δ that uses the state vector s of the insight k. Finally, the augmented network drives the overall network O that approximates $p_u(k)$ shown in Equation 3.

Insight	Are you interested to see more of these type of insights?
On Weekdays you walk less than on Weekends	◯
Your heart rate is high on Mondays than other days	◯
when you bike, you spend less calories per minute than when you run	◯

Table 2: A Sample insight feedback form

$$p_u(k) \sim O(N^{'}(d_1, d_2), \Delta(s)) \tag{3}$$

The neural network learns the function p_u by minimising the mean squared error loss function J_2 as shown in Eq 4.

$$J_2(\theta) = \frac{1}{n}\sum_{i=1}^{n}(p_u(k) - O_\phi(N^{'}(d_{1i}, d_{2i}), \Delta(s_i))^2 \tag{4}$$

In this work, we show that any improvement in approximating p_u does not have an impact on the approximation of f in Equation 1.

3.2.2 User Model Acquisition

The online learning strategy detects more interesting insights without being instructed by the user explicitly. It uses a feedback form in a mobile application that displays a few insights that were scored high by the base model. The users may choose the insights that they are interested in and the system learns from it. A sample feedback form is shown in Table 2. In this work, we simulated the user preferences to change every month as its tracking is a problem by itself.

This feedback is equivalent to "labeling" in traditional online learning theory. To generate the insights so that our online learning system can be validated, we obtained sleep and environmental sensor data collected from a bedroom of a volunteer over a period of 4 months from May 2019 to August 2019. We logged various parameters such as the timestamp of the start of sleep, sleep duration, sleep latency, ambient light, ambient temperature, ambient sound and timestamp of waking-up. We generated insights for each day of the user using the procedure explained in (Härmä and Helaoui, 2016). The insight texts talk about the two contexts that it compares and an expression of the comparison. The number of insights per day varied between a few hundred to few thousand. We simulated the user preference given below by automatically filling the feedback form for each day.

1. May: The user is interested in Insights related to Weekdays.
2. June: Weekend insights are interesting to the user.
3. July: The user prefers to know more about his sleep duration.
4. August: The user is again interested to know if he/she is doing well on weekends.

All statistically significant insights per day on a given month that satisfy the corresponding preference criteria were labeled with interestingness score 1 and otherwise labeled 0. Since neural networks understand only numbers, we encoded each comparison insights into a single dimension binary vector s containing 220 elements where each element correspond to one parameter of comparison. For example, one element corresponds to each day of the week. Hence, if the comparison is related to Mondays and weekends, the elements corresponding to Mondays, Saturdays, and Sundays are assigned a binary one and the rest are assigned zero. We inject this vector to the model while transfer learning for interestingness recognition. In the following subsection, we explain how the model is transfer learned and how the online learning pipeline is implemented and evaluated.

3.2.3 Transfer Learning

Transfer learning was performed to enable the model to learn insight interestingness in addition to significance. The self-learned model was frozen from the input layers up to and including the F5 layer. The vector s is passed as input to another fully connected layer F6 with 100 neurons. This layer is concatenated with the F5 layer as shown in Figure 2. The concatenated layers are fed to another fully connected layer F7 having 100 neurons. While the layer F6 is linearly activated, the F7 layer is activated by the ReLu function. Finally, the output layer is a single neuron fully connected layer activated by a sigmoid activation function. Notice that the final layer is activated by a sigmoid function as this is a binary classification problem trained on user preferences instead of significance.

By performing this transfer learning, the model retains the features that correspond to the significance and simultaneously recognise interestingness of insights based on user preference.

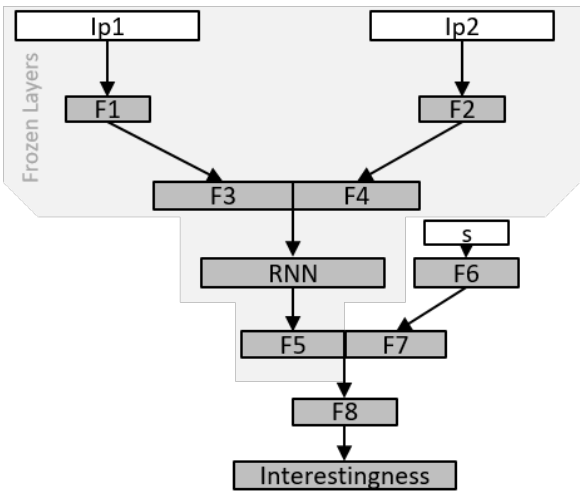

Figure 2: Augmenting the base network for online learning

3.2.4 Learning Modes

The architecture of the online learning scheme is presented in Figure 3. The scheme is executed in two modes, namely, accelerated learning mode and normal learning mode. These modes determine how much the models are trained (iterations). The accelerated learning mode, by default, starts from the first usage of the insight generator for the first ten days. Then, the normal mode begins. During the accelerated learning mode, the model learns more rigorously and during the normal mode, it learns at a normal phase. This is achieved by varying the number of iterations of training for each day. This the accelerated training mode has more iterations of training.

3.2.5 Training and Validation Switch Logic

Every day, the insights are assigned an iterestingness value based on user feedback and are scored by the model. Based on the learning modes, two scenarios can happen that impacts whether the insights are used to train or validate the model.

1. If the system is in accelerated training mode and the insight has a prediction error of less than 0.3. The training and validation switch pushes a copy of the insight to both training and validation pools. Therefore, the model trains and validates these insights.
2. If the mode is the normal training mode
   - If the prediction error is less than a preset threshold (0.10) and 50% random chance is satisfied and the fraction of interesting insights in the validation pool (if updated) will be between 0.42 to 0.6, the switch pushes the insight into the validation pool.
   - Else, if the percentage of interesting insights in the training pool (if updated) will be between 42% to 60% (arbitrarily chosen), the switch pushes the insight into the training pool.

If the user does not give any feedback, the insights continue to get pooled and trained based on the older feedback. This implicitly assumes that the user's preference is unchanged. However, we allow a small error to occur so that the system also has the ability to pick other insights at times instead of strictly catering to the user preference.

3.2.6 Pool Maintenance Logic

Both the pools are maintained to hold only a maximum limit of days of data. We fixed this arbitrarily to be 14 days. Here we assume a user's interestingness remains fairly unchanged for a period of two weeks. Every 20 days, the model forcefully pops 7 days of data in a FIFO fashion. This helps to avoid overloading the training and validation pools and forgetting older preferences. Additionally, the validation pool is completely emptied at the beginning of the first day of the normal learning phase.

3.2.7 Update Logic and Metrics

At the end of every day, a copy of the model is trained on the training pool and validated on the validation pool. If the validation accuracy exceeds a set limit (here 70%), the old model is replaced by the recently trained model. However, as an exception in the accelerated learning mode, the model is updated every day irrespective of its performance. This purposefully over-fits the model to the insights during accelerating learning mode. The performance of online learning is monitored using statistical measures, namely, sensitivity, specificity, and accuracy in predicting the interestingness of insights. Additionally, we introduce the significance preservation score, which is calculated as shown in Equation 5.

$$P_s = N_a/N_p \tag{5}$$

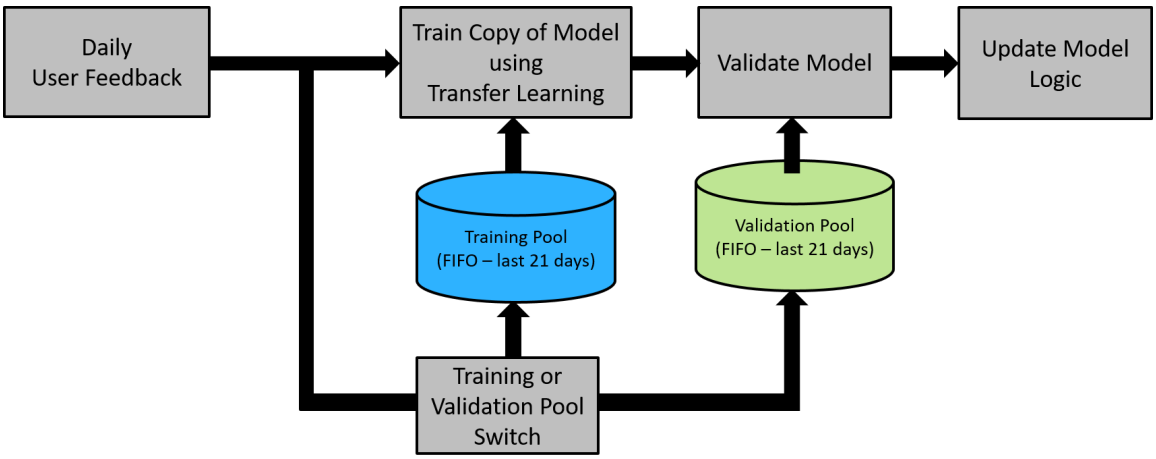

Figure 3: Online learning through user feedback

where, N_a and N_p are the number of actual interesting insights in the validation pool and the number of predicted interesting insights during validation, respectively. The P_s is not defined when N_p is zero. This is a limitation of the metric.

4 Experimental Results ad Discussions

In this section, we present the results that we obtained at each stage.

4.1 Choosing The Base Model Architecture

An example of histograms of significant and insignificant pairs of normal distributions is shown in Figure 4. It also demonstrates the variation of magnitude, range and cardinality (more samples have a smoother curve) of the synthetic data. Each of the base model architecture, namely, RNNA, RNNB, and SIAM were Trained, validated and tested using the dataset containing only normal distributions. The performance of each model is presented in Table 3. We observed that the RNNA model exhibits a test accuracy of 92% in predicting whether an insight is interesting or not. The performance of RNNA is thereby comparatively better than that of RNNB. This shows that more neurons do not always lead to improved performance. Also, RNNA exhibits slightly better performance than the SIAM network. This could be due to the sequential treatment of the data by the RNN which is part of the network. Additionally, since the SIAM network has fewer neurons, it also provides evidence that lesser neurons might not help either. In our view, the neural model should have an adequate number of neurons and parameters and an explainable architecture, which is, unfortunately, missing in recent works in this field. Hence, the RNNA architecture is chosen as the base model and considered for further analysis.

Table 3: Performance of different models while training and testing with normal distribution

MODEL	DESCRIPTION	ACCURACY
$\alpha = 0.05$		
RNNA	BIDIRECTIONAL RNN LAYER	0.92
RNNB	MORE NEURONS	0.86
SIAM	SIAMAESE NETWORK	0.87

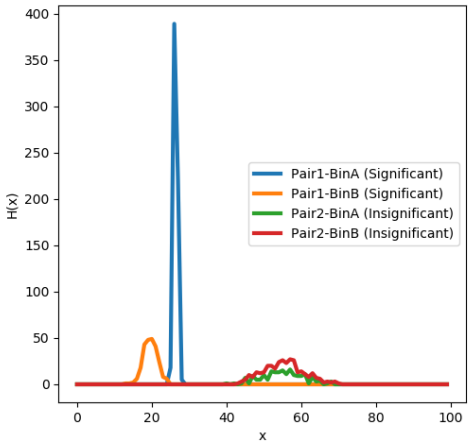

Figure 4: Pair of normal distributions without significant difference

4.2 Improving Base Model Training

We trained the base model using diverse pairs of distributions (histogram) such as Gamma, Gumbel, Laplace, Normal, Uniform and Wald. We observe that when we tested each distribution as shown in

Figure 5, we find out that the performance of the model to normal distribution remained at 0.92, but the uniform was even higher at 0.97. The worst performance was observed on Wald distribution. We have additional evidence that this is a limitation of the actual KS test that is being reflected in the neural model. It is also found that few distributions exhibit improved performances as alpha increases and few showed weaker performance as alpha increases.

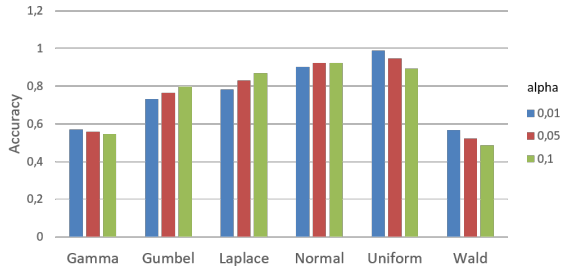

Figure 5: Gaussian trained model on mixed distributions

4.3 Online Learning

We initiated the online learning scheme and the performance metrics are presented in Figure 6. We stated the system in the accelerated learning mode for the first 10 days. It is observed that the accuracy, sensitivity, and specificity were unstable during the first 4 days of the accelerated learning phase. From the fifth day onwards, the three measures show improvement and are in the range of 0.9 to 1. The P_s measure is not defined when there are no significantly valid insights that are interesting. This is observed till day 3 and on Day 4, 100% P_s is observed. This implies that the model exhibits significance preservation starting at least from day 4 onwards. The performance is rather stable all the while during the remaining days of May and the entire June. Even though there is a transition between weekday insights and weekend insights, the model seems to adapt very well. In the months of July and August, there are visible drops in the performance around the 10^{th} day of the month even though the preference changed on the 1st of both months. This could be an instability caused due to the sudden rise in the training pool and reduction of validation pool data as shown in Figure 7. In General, the pool maintenance logic is able to control the number of training and test data points. Although the first half of July saw a huge influx of training data, the maintenance logic prevented the training pool from overloading. Otherwise, there would have been a huge chance of exposing the model to noise in the data. The mean squared error (MSE) curve shows that the error between predictions and ground truth is not very high. The MSE decreased more steeply during the accelerated learning mode compared to the normal mode. There are periodic valleys in the training pool count and validation pool count denoting the reach of the 20-day window for cleanup of the pool. Also, additional cleanups are done every day when the number of days of insights in the pool exceeds 14. All cleanups on the training and validation pool are indicated by fain red vertical lines in Figure 7.

5 Conclusions and Future Scope

In this work, we propose a neural model capable of learning the Kolmogorov-Smirnov statistical significance test and we augment architecture to learn user preference with an online-learning scheme. To model statistical validity tests, we chose a base neural model, for which three architectures, namely, a simple neural network with recurrent neural network layers with fewer neurons, similar networks with more neurons and a slightly different siamese network were investigated. The neural network with the recurrent neural network layers having lesser neurons exhibited the best performance. We continued to develop a smarter network that can not only identify an insight but also learn its interestingness in an online setting. For this, we used transfer learning and online learning approaches. We froze a part of the base model and augmented it with an additional input layer that reads a binary filter vector that describes an insight. We trained it on a real dataset while simulating user preference. The model was generally stable with few transients when the user preference changed. We were able to show that the model preserved its knowledge about statistical significance while learning interestingness. This made the network unique in an intelligent way as this is the first attempt, that a single neuron could perform more than one functionality. In the future, we would like to test the capability of the online learning module in a scenario where user preference can take multiple states at the same time.

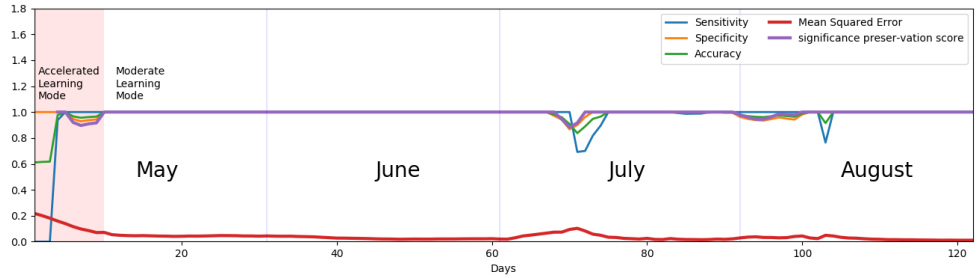

Figure 6: Timeline of Online Learning with Performance Indicators

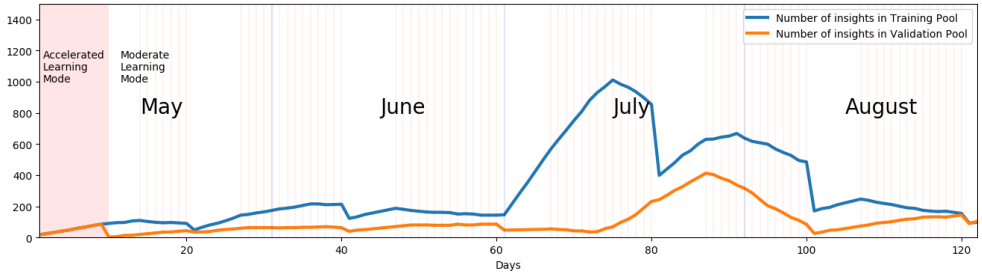

Figure 7: Size of Training and Validation Pool

Acknowledgments

This work was supported by the Horizon H2020 Marie Skłodowska-Curie Actions Initial Training Network European Industrial Doctorates project under grant agreement No. 812882 (PhilHumans).

References

Rakesh Agrawal and John C Shafer. 1996. Parallel mining of association rules. *IEEE Transactions on knowledge and Data Engineering*, 8(6):962–969.

Harm op den Akker, Miriam Cabrita, Rieks op den Akker, Valerie M Jones, and Hermie J Hermens. 2015. Tailored motivational message generation: A model and practical framework for real-time physical activity coaching. *Journal of biomedical informatics*, 55:104–115.

Samuel Berlemont, Grégoire Lefebvre, Stefan Duffner, and Christophe Garcia. 2015. Siamese neural network based similarity metric for inertial gesture classification and rejection.

Daniel Braun, Ehud Reiter, and Advaith Siddharthan. 2018. Saferdrive: An nlg-based behaviour change support system for drivers. *Natural Language Engineering*, 24(4):551–588.

Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. 1996. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37–37.

Alex A Freitas. 1999. On rule interestingness measures. In *Research and Development in Expert Systems XV*, pages 147–158. Springer.

Aki Härmä and Rim Helaoui. 2016. Probabilistic scoring of validated insights for personal health services. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–6. IEEE.

John P Higgins. 2016. Smartphone applications for patients' health and fitness. *The American journal of medicine*, 129(1):11–19.

Melanie Hingle and Heather Patrick. 2016. There are thousands of apps for that: navigating mobile technology for nutrition education and behavior. *Journal of nutrition education and behavior*, 48(3):213–218.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I Jordan. 2015. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Proceedings of the 1st Workshop*

on Representation Learning for NLP, pages 148–157.

Samuel J O'Malley, Ross T Smith, and Bruce H Thomas. 2012. Data mining office behavioural information from simple sensors. In *AUIC*, pages 97–98.

Ehud Reiter, Roma Robertson, and Liesl M Osman. 2003. Lessons from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144(1-2):41–58.

Burr Settles. 2009. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences.

Burr Settles. 2011. From theories to queries: Active learning in practice. In *Active Learning and Experimental Design workshop In conjunction with AISTATS 2010*, pages 1–18.

Somayajulu G Sripada and Feng Gao. 2007. Linguistic interpretations of scuba dive computer data. In *2007 11th International Conference Information Visualization (IV'07)*, pages 436–441. IEEE.

Nandan Sudarsanam, Nishanth Kumar, Abhishek Sharma, and Balaraman Ravindran. 2019. Rate of change analysis for interestingness measures. *Knowledge and Information Systems*, pages 1–20.

Jie Tao and Xing Fang. 2020. Toward multi-label sentiment analysis: a transfer learning based approach. *Journal of Big Data*, 7(1):1–26.