UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Alo Aasmäe

Benefit prediction of buying Customer Relationship Management features of Pipedrive

Master's Thesis (30 ECTS)

Supervisor(s): Gamal Elkoumy, MSc
Roland Kriibi, BSc

Tartu 2022

Benefit prediction of buying Customer Relationship Management features of Pipedrive

Abstract:
People that use software such as CRM applications, which enables to track relationships with their clients, have always been interested in knowing the future prospects of their clients, especially when additional value could be offered to the client. Applying process mining techniques in combination with adversarial model training to make these predictions have been scarcely done in the context of CRM applications to predict user behavior. By defining an event that is assumed to have an effect on user actions, it is possible to split a user's event logs into two parts based on the timestamp of this event, enabling to create a mapping between the user's actions before and after. This thesis applies both process mining and machine learning on event logs to predict future user actions based on previous user behavior. Here we show that this approach is viable in a business context by using event logs extracted from Pipedrive users and that the solution could provide value in marketing scenarios. The result is an automated way of predicting user metrics for custom "what-if" scenarios, provided that sufficient event logs are present. The effect of this is potentially beneficial for both the end-user and employee of Pipedrive, by enabling a way for a user to see the effects of further investment and providing a better direction for allocating resources in marketing.

Keywords:
Customer Relationship Management, Process Mining, Deep Learning

CERCS:
P176 Artificial intelligence

Pipedrive'i CRM toodete ostmise kasu ennustamine

Lühikokkuvõte:
Inimesed, kes kasutavad CRM süsteeme, on alati olnud huvitatud enda klientide käitumisest tuleviku perspektiivis, eriti kui see on seotud rahaliste tehingutega. Kasutaja käitumise ennustamist kasutades protsessikaeve tehnikaid koostöös võistleva mudeli treenimisega CRM rakenduste kontekstis on vähe uuritud. Valides välja sündmuse, millel kohta arvatakse, et sellel on mõju kasutaja tegevustele, on võimalik kasutaja sündmuste logid jaotada kaheks osaks selle välja valitud sündmuse ajatempli põhjal, mis võimaldab luua kaardistuse kasutaja tegevuste vahel enne ja pärast sündmust. Antud lõputöö rakendab protsessikaevet ja masinõpet sündmuste logide pea, et ennustada tulevasi kasutaja tegevusi eelneva käitumise põhjal. Selle tööga näitame, et pakutud lähenemine on rakendatav ärikontekstis, kasutades sündmuste logisid, mis on eraldatud Pipedrive'i kasutajatelt. Lisaks näitame, et lahendus lisab väärtust erinevates turundamise stsenaariumites, kogudes tagasisidet turundusosakonnalt lahenduse kohta. Töö tulemuseks on automatiseeritud viis kasutaja meetrikate ennustamiseks kohandatud stsenaariumite jaoks,

eeldusel, et selleks tarvilikud sündmuste logid on saadaval. Selle töö mõjuna on kasu potentsiaalselt nii Pipedrive'i lõppkasutajal kui ka selle töötajatel - ühelt poolt on kasutajal võimalik näha enda edasiste investeeringute mõju ning töötajatel on võimalik teha veelgi kaalutletumaid otsuseid turundusega seotud teemadel.

Võtmesõnad:
Kliendisuhete haldus, Protsessikaeve, Süvaõpe

CERCS: P176 Tehisintellekt

Contents

Glossary

event label Label of a corresponding activity being executed, e.g. the activity of a business process.

event log Description of an event which has at least two attributes: the label of the event and its timestamp.

event prefix A partly-complete event trace followed by an event suffix.

event suffix A continuation of an event prefix.

event timestamp Time indicating when the event has been recorded.

event trace A sequence of events, ordered by the event timestamp.

process discovery A set of techniques that construct a representation of an organisation's current business processes.

process mining A family of techniques to support the analysis of operational processes based on event logs.

schema Description of the structure and organization of data in a database system.

unicorn A company valued at $1 billion or higher.

Acronyms

BPMN Business Process Model and Notation.

BPS Business Process Simulation.

CRM Customer Relationship Management.

DL Damerau-Levenshtein.

DW Data Warehouse.

FSA Finite State Automaton.

ILP Integer Linear Programming.

LDAP Lightweight Directory Access Protocol.

MLMME Maximum Likelihood Min-Max Estimation.

PM Product Manager.

PM^2 Project Management for Process Mining.

RE Regular Expression.

S3 Simple Storage Service.

SaaS Software as a Service.

VPN Virtual Private Network.

1 Introduction

CRM software enables the users to track relationships with their clients. An often sought out feature is knowing the future prospects of their clients, especially when the user has an opportunity to provide additional value to them.

A way to approach this problem is by using process mining, which is a way of extracting information from event logs that are recorded by an information system, such as a CRM system, of an organization [vdAW05]. Event logs record the execution of activities within the organization. This thesis applies process mining techniques in combination with adversarial model training to predict future user actions based on previous user behavior in a CRM application.

1.1 Pipedrive

Pipedrive is a cloud-based SaaS company founded in 2010, most known for its Customer Relationship Platform, Pipedrive[1]. In 2020 the company sold its majority to Vista Equity Partners, an global investment firm, and became the fifth unicorn of Estonia [Han20].

Pipedrive has many different views based on the intended action the user desires to make with their clients' contacts or organisations, but the base of it all has always been the pipeline view of deals, as seen on Figure 1. A deal is a series of interactions with the intent of reaching a desired conclusion by the user and is the main object that connects the user of Pipedrive with their clients.

The pipeline gives an overview of the different stages a deal can be in. A stage is a descriptive status of a deal with the intent of organizing deals into logical subgroups. These stages are fully customisable by the user, but a deal is intended to resolve into one of two states - won or lost deals. There is also an option that the deal stays unresolved for a longer period of time with no resolution, but in the scope of this thesis this is interpreted as a deal that is still in progress.

1.2 Motivation

Pipedrive is in a unique position where it is in the process of going from a single product company (sales software as a CRM) to a multi-product company. The first of these new products is a marketing tool called Campaigns by Pipedrive, presented on Figure 2, designed for creating email campaigns with unified contact lists between users of a company and customisable email templates[2].

With every new product, Pipedrive has to make an investment to a new area for them, which brings about a certain amount of risk. Moreover, once a new product has been

[1] `https://www.pipedrive.com/en/about`

[2] https://www.pipedrive.com/en/features/campaigns

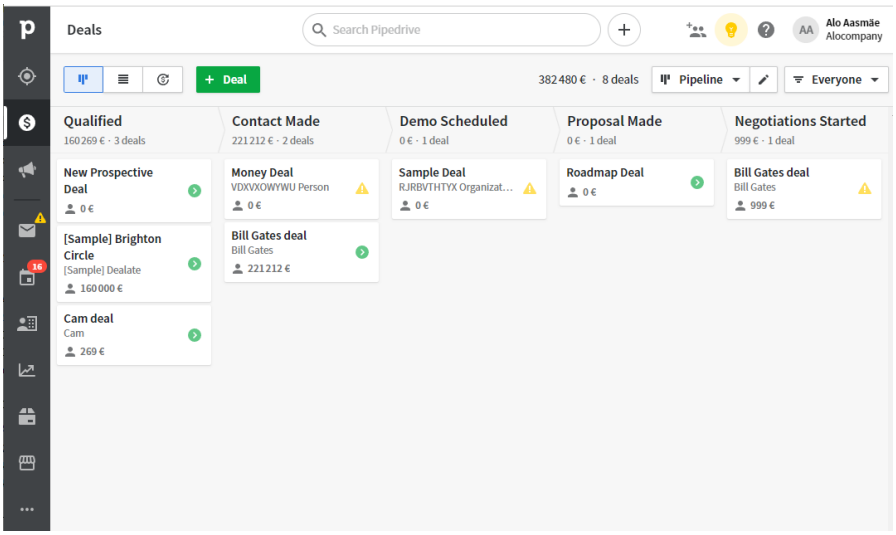


Figure 1. Example of Pipedrive pipeline.

developed and released, the users of existing products have to decide whether a new product is worth the additional investment in both money and increased manpower.

If there existed a way of predicting how adding a feature would affect the users' desired metrics, it would help customers to maximize the benefits of using Pipedrive and with that increase the brand loyalty. The company would gain confidence that their investment will be worthwhile and the users would get immediate feedback what features would suit them the best, whatever metric should they desire to improve.

1.3 Contribution

Process mining can be used to address the above problem. The input to process mining techniques is the so-called *event log*. An event log is a record of an event, in this case user activities in Pipedrive, which has at least two attributes: the label of the event and its timestamp. To alleviate the above problem, an approach that combines process mining of event logs and machine learning is taken. The aim of this thesis is to create a machine learning model for predicting new user behavior based on existing user behavior, where the behavior is extrapolated from the user's event logs in Pipedrive. Part of creating this model includes extraction and preparation of event traces, building and training of the model, and finally applying the model to predict new user events. Furthermore, a survey is created as an evaluation for the trained model to collect feedback from the marketing department of Pipedrive about the potential of this solution. The results of the survey are then analyzed to see whether the solution is viable for future development and aligns with the business processes of Pipedrive.

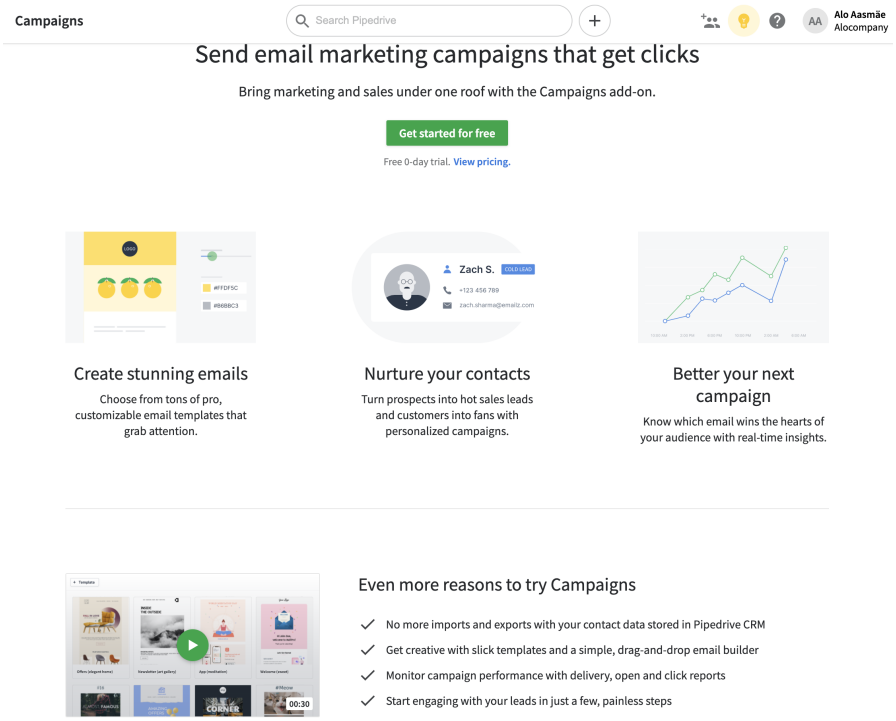


Figure 2. Pipedrive Campaigns view.

The two contributions are formulated into two research questions, stated in chapter 1.4.

1.4 Research questions

In this thesis, we provide a method to predict the benefit of adding a feature to a user's account within Pipedrive CRM system. Specifically, we address the following research questions:

RQ1: Given an event log from the CRM system of Pipedrive, can we use the deep adversarial model to predict the benefit of adding a feature to the user's account?

RQ2: Do the results of the proposed approach provide (promotional) value to Pipedrive marketing department for promoting their products?

1.5 Thesis outline

This thesis consists of seven chapters. Chapter 2 gives an overview of related work and advancements in the field of process mining and adversarial training. Chapter 3 is dedicated to elaborating on the data extraction and preparation for model training.

Chapter 4 goes more into depth about the approach and implementation of the predictive machine learning model. Chapter 5 elaborates on the feedback collection about the implementation from key related people in Pipedrive.

Chapter 6 expands on the results of both the performance of the model and the feedback gathered from Pipedrive employees. Chapter 7 concludes the thesis, gives a summary on what was achieved and also supplies some notes on potential future work with this topic.

2 Related work

This chapter gives an overview of definitions related to process mining and presents the current approaches to predict user's behavior.

2.1 Definitions

Process mining is a way of extracting information from event logs that are recorded by an information system [vdAW05]. An event log is a record of an event, which has at least two attributes, the label of the event and its timestamp.

Definition 1. Given an event label a_i and a timestamp t_i, an event log is a tuple $e_i = (a_i, t_i)$.

Table 1 shows an example of an event log in Pipedrive that occurred on Jan 1 2022, where the contents of the event is adding a deal.

event_name	time
deal_added	2022-01-01T11:00:33

Table 1. Example of an event log.

Events that we are looking at are always a part of some business process that can consist of one or more sequence of events, referred to as an event trace. Event traces are usually ordered by the timestamp of its event logs.

Definition 2. An event trace σ is a sequence of n events, $\sigma = \langle e_1, e_2, \ldots, e_n \rangle$.

A set of a user's business processes in a system is a collection of event traces generated by the activities that the user has done, also called a dataset of event logs.

Definition 3. A dataset of event logs L is a set of i event traces $L = \{\sigma^{(1)}, \sigma^{(1)}, \ldots, \sigma^{(i)}\}$.

An event trace can also be viewed as two separate sets of ordered traces that are split based on some condition, for example the timestamp of events. These two sets are called the event trace prefix and suffix.

Definition 4. Given an event trace $\sigma^{(i)} = \langle (a_1, t_1), (a_2, t_2), \ldots, (a_m, t_m) \rangle$ of some dataset L, the prefix of events of length k for event trace $\sigma^{(i)}$ is defined as $\sigma^{(i)}_{\leq k} = \langle (a_1, t_1), (a_2, t_2), \ldots, (a_k, t_k) \rangle$.

Definition 5. Given an event trace $\sigma^{(i)} = \langle (a_1, t_1), (a_2, t_2), \ldots, (a_m, t_m) \rangle$ of some dataset L, the suffix of events that corresponds to a prefix of events of length k for event trace $\sigma^{(i)}$ is defined as $\sigma^{(i)}_{>k} = \langle (a_{k+1}, t_{k+1}), (a_{k+2}, t_{k+2}), \ldots, (a_m, t_m) \rangle$.

2.2 Generating Event Logs through the Simulation of DECLARE Models

A common approach for testing different process discovery algorithms is using synthetic logs created via simulation, which allow researchers to have more control over the data, as opposed to already existing publicly available logs, that may contain imperfect information [CM15].

XES (eXtensible Event Stream) has been developed as a standard of storing, exchanging and analyzing event logs, where an event refers to an activity and is related to a particular process instance. When these events are ordered within a case, it can be seen as one execution of the process or event trace.

Although model simulators and log generators had been present prior to the work above that generate synthetic logs through the simulation of a procedural process model, it is not suitable for process discovery techniques based on declarative business models. The authors claim that tools for generating event logs based on the simulation of declarative models are not available in the literature, which was the problem that the paper above tried to address.

The process models defined in the work above are meant to be defined using DECLARE, which sets the base of models on a set of constraints exerted on sets of activities, which must hold true during process enactment. This type of approach states that what is not explicitly defined is allowed, allowing greater flexibility to describe processes working in unstable environments.

DECLARE specifies an extensible set of standard templates that a process analyst can use to model a process. These templates can be divided into two main groups:

- Existence templates or unary templates, expressed as predicates over one variable
- Relation templates which comprises rules that are imposed on the target activities

The first step in the process for generating event logs is mapping activity names of events to single terminal characters that can be used with templates. The templates themselves represent constraints that are expressed on the activities of the process. The constraints can be expressed as regular expressions (REs). Do note that the authors use REs as translations of business rules, and not as production rules for the topology of the process.

For each RE, a Finite State Automaton (FSA) is derived and with FSAs it is possible to get the products. The constraints representing the process behavior, and the conditions on the length of traces must hold true at the same time. Traces of the event log are created on the basis of the strings accepted by the automaton. This is done by choosing a random path along the automaton that terminates in an accepting state.

The result is a string made of characters that uniquely identify activities, in a sequence that complies with the constraints of the input model M. The corresponding activities from each character of the generated strings is retrieved using the inverse translation

mapping to the single-character identifiers. The procedure is repeated user-defined N times.

The effectiveness of the resulting tool was evaluated in two ways. The first is using a plug-in called DECLARE Miner to mine a log of a process and checking whether the discovered DECLARE model was in line with the simulated one. The second way is attempting to use the tool to reproduce the behavior of a real-life log.

The authors concluded that the logs generated by their approach reproduce exactly the behavior of the input models. This was regardless of the discovery algorithm adopted to verify it, and irrespectively of whether the aim was to simulate a hand made reference model, or rather to replicate the behavior of an existing real life event log.

The result of the above paper is a tool for log generation based on DECLARE models, the success of which is backed up by successful experiments with high results. The approach relies heavily on data structures and templates defined by DECLARE, which could create unnecessary overhead in attempts of generalization.

2.3 Generating Synthetic Event Logs based on Multi- perspective Business Rules

Process discovery techniques and tools require the availability of event logs, but often the real log files contain noisy data and do not have some given characteristics required to execute discovery techniques. Declarative process models allow flexibility in describing highly variable behaviours. One of the main challenges in the context of testing with declarative models is the capability of supporting multi-perspective specifications [HM16].

This is the reason a tool was developed for log generation based on multi-perspective DECLARE models. The tool allows the user to generate logs with predefined characteristics, for example defining the number and length of the process instances, which is compliant with a given DECLARE model.

The IEEE Task Force on Process Mining has recognized XES (eXtensible Event Steam) as a standard for event logs. Business Process Management Systems (BPMS) have the ability to store information about workflow execution in the context of these event logs. In the thesis above, process models are defined using DECLARE process modelling language. DECLARE also supports data-flow, meaning applying a constraint only when data meets the requirements. The thesis uses `lp_solve` to solve integer linear programming (ILP) problems [3].

The approach proposed in the thesis above for log generation is as follows:

1. Split Activation Constraints - The main goal of this step is to split activations based on validity of the condition and generate all possible combinations without any duplications.

[3] `http://lpsolve.sourceforge.net/5.5/`

2. Alphabet Cleaning - Using ILP to remove elements from the alphabet that cannot be true together. This is the place where the main contribution of the thesis comes from.

3. Generate MINERful Input Model - MINERful is a tool used to generate event logs [CM15]. For this three inputs are required:

   - the input model
   - minimum and maximum size of events per trace
   - total number of traces that the generated log must contain

4. Restore Keys and Add Data - The log generated from MINERFul consists of characters that differ from the actual activity name of the process model. In this step, original names of these activities without altering their traces and events order is restored.

5. Events log with data - This is the final step of log generation. The generated log with data is stored at a location specified by the user and in a user selected format i.e., XES format. To complete this task log storing services provided by MINERFul had been used.

This thesis addressed the capability of supporting multi-perspective specifications in the context of testing with declarative models. The solution proposed consisted of developing a tool for log generation based on multi-perspective DECLARE models.

2.4 PLG: A Framework for the Generation of Business Process Models and Their Execution Logs

The paper above presents a problem involving business models that are created through process mining being hard to evaluate due to lack of data. A way around this problem is to try to generate "realistic" business process models together with their execution logs [BS11].

To address this problem, the authors proposed a new tool called the "Processes Logs Generator" which serves two purposes. The first is to generate a random business process through user-defined parameters. Secondly, this tool can also be used to execute the generated processes and register each executed activity.

Previous works related to this problem have used the approach of generating a Petri net that represents processes, but the approaches have not addressed the problem of generating traces from the developed Petri nets [HL10].

The authors decided to adopt a general formalism of a business process model description through dependency graphs in order to appeal to an inexperienced user not

used to refining Petri nets. The dependency graph is defined as a graph consisting of vertices, which represent an activity of the process, and edges representing a dependency relationship between any given two activities.

The strategy adopted for the generation of a process is based on the recursive composition of basic patterns introduced to represent different patterns that translate well into a grammar. These patterns are:

1. The direct succession of two workflows;
2. The execution of more workflows in parallel;
3. The mutual exclusion choice between some workflows; and
4. The repetition of workflow after another workflow has been executed ("preparing for the repetition)

In order to give control on the complexity of the generated processes, a probability to each production in the grammar is introduced. This allows user defined parameters to control the probability of occurrence into the generated process of a specific pattern.

The whole procedure of using the grammar has been implemented in a tool developed in Java. The tool consists of two main components, a library and a visual tool for the generation of one process. With this tool, the user can control the probabilities of grammar productions, as well as the maximum depth of a grammar production, meaning how many times can the production be nested before it has to reach a terminal.

The above work addressed the problem of evaluating business process logs with random generation of business processes and their execution logs. The authors claim this to be a first step in addressing the problem of random generation of business processes, but the proposed solution does not take into account details of a specific business. It is just a way of generating a random business model with parameters that can shift the randomness in some way.

2.5 Simod: A Tool for Automated Discovery of Business Process Simulation Models

Business process simulation (BPS) model creation is traditionally done through manual labor, via interviews, observation and sampling, which makes the traditional approach time-consuming and error-prone. Process models produced by domain experts generally do not capture all possible process execution paths [CDR19].

Previously proposed approaches have generally assumed that the event log perfectly fits the process model given as input, which is not usually the case. The paper above presents a tool named Simod to automatically generate simulation models from event logs. The tool implements pre-processing techniques to ensure conformance between

event logs and the discovered models, plus a hyperparameter optimization technique to fine-tune the accuracy of the resulting model.

The above work the first version of Simod, which automatically creates simulation models that can be executed using the BIMP simulator[4].

The steps of creating a process simulation model are as follows:

1. Pre-processing - extracting a BPMN model of the business process from data and guarantees its quality and coherence with the event log. This consists of:
   (a) Control Flow Discovery - Mining the BPMN model using solely an event log in XES format as input;
   (b) Trace alignment - Simod measures the conformance between the process model and the event log, and provides the option of managing the non-conformances by removal, replacement or repairing.
2. Processing - Simod extracts the simulation parameters and assembles them with the process structure to create a BPS model.
   (a) Parameters Extraction - In this step, all simulation parameters are calculated;
   (b) Simulation Model Assembly - In this step, simulation parameters are merged with the BPMN model into a single data structure.
3. Post-processing - Simod measures the similarity between the results of the generated simulation model to the original event log. The simulation model similarity assessment is carried out using the Damerau-Levenshtein (DL) algorithm with a modification which includes a time penalty.

The above work addressed the problem of automated discovery of business process simulation models. For this, a tool to automatically generate simulation models from event logs. Instead of generating the event logs directly, a business model is created from which the synthetic event logs can be extracted from.

2.6 Customer Journey Analysis at Pipedrive: A Process Mining Approach

Pipedrive relies mostly on its Product Managers (PMs) to know the system, its requirements and workflows, with each PM focusing on some part of the Pipedrive application [Aut21]. On one hand, this takes advantage of developing specialists that have gained a deep knowledge of a field over a period of time and leverages itself on knowledge sharing within the company. On the other hand, this creates a situation where (PMs) are

[4] https://bimp.cs.ut.ee/

required to conduct user surveys and analyse usage reports to get an overview of the customer journey.

To approach this problem, the above work proposed two research questions:

1. What distinguishes customer journeys between trialled customers that convert to a paid account versus those who abandon?

2. Are there any particular roadblocks or bottlenecks that prevent trialled customers from converting?

Furthermore, the author used a modified version of a methodology called Project Management for Process Mining (PM^2), focusing on problem framing and data collection to conduct process mining and analysis [vELLvdA15]. This involved working with event logs in Pipedrive with process mining tool Apromore[5].

The findings from the thesis are addressed by resolving the stated research questions as follows:

- A Pipedrive trial user who uses the system consistently is most likely to convert to a paying customer;

- A higher degree of customization of Pipedrive leads to the customer being able to tailor the system to their needs;

- Users that required more assistance about using the system were more likely to abandon the trial or not convert to a paying customer.

2.7 A Deep Adversarial Model for Suffix and Remaining Time Prediction of Event Sequences

Taymouri et al. focuses of event suffix and remaining time prediction, which are addressed as sequence to sequence learning tasks [TLRE21]. An encoder-decoder framework is proposed to learn a mapping from a set of event prefixes to a set of based on the traces the events belong to. The result of this is open-loop training where the events are time-dependent.

In addition to this, since open-loop training does not capture the joint temporal dynamics of events in ground truth suffixes, adversarial training using Generative Adversarial Nets is used to learn the joint probability distribution of events suffixes, given their respective event prefixes.

The proposed framework for the approach in the work above consists of three parts:

1. Data preprocessing is done to prepare the input data to the for of event prefixes and suffixes;

[5] https://apromore.com/

2. An adversarial predictive model called Maximum Likelihood Min-Max Estimation (MLMME). is created which consists of two neural networks, the generator and the discriminator. The former provides predictions and the latter evaluates the predictions using ground truth. This way, the generator receives feedback from ground truths directly and also through the discriminator's output.

3. Finally, beam search is used to have more than one suffix for an event prefix.

The resulting framework was evaluated on four real-life publicly available datasets[6][7][8]. To evaluate the performance of suffix prediction, Damerau-Levenshtein (DL) distance was used [Dam64].

Two main results were concluded:

1. the approach proposed in the paper above learns the temporal dynamics of sequences more accurately than their respective baselines;

2. the proposed training method also boosts a model's performance to capture the temporal relationships of events in sequentially related data.

[6] https://data.4tu.nl/articles/dataset/Dataset_belonging_to_the_help_desk_log_of_an_Italian_Company/12675977

[7] https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204

[8] https://data.4tu.nl/articles/dataset/BPI_Challenge_2017/12696884

3 Data collection and preprocessing

As stated with RQ1 in Chapter 1.4, one of the objectives of the thesis is to create a model that takes a set of partial event traces, also known as the event prefixes, as its input and predicts the events that are to complete the event trace, also known as event suffixes.

The approach is going to be based off the framework proposed in Chapter 2.7. For this, data from Pipedrive is needed as input. This chapter is dedicated to understanding the data being worked with throughout the process, where it originates from and what sort of data is deemed suitable for this task.

3.1 Organization of data in Pipedrive

The primary data objects that users use from a business perspective in Pipedrive are as follows[9]:

- **Organizations** - Organizations represent the company or entity that the customer of a Pipedrive user is related to. An organization can have multiple People related to it along with multiple Deals.
- **People** - People represent the "Customer" part of CRM. These are the contacts that the user of Pipedrive are directly in contact with related to a deal. A People contact can be linked to only one Organization, but can have multiple Deals, Activities and Mails associated with them.
- **Deals** - Deals represent the ongoing transaction the user of Pipedrive is pursuing with a Person or Organization. Deals can be linked with a Person or an Organization, or both. Activities can also be scheduled in relation to a Deal. Every Deal is considered ongoing, until it is either Won or Lost.
- **Activities** - Activities represent actions that the user of Pipedrive can schedule and undertake with Organizations, People, and Deals. Associating an Activity with a Deal will also associate the Activity with the Deal's other related objects, either with the respective Person and/or Organization.
- **Leads** - Leads can be considered as a precursor to Deals, representing a potential sales process that is not ready to be worked on for any reason. A Lead has to be related to an Organization or Person, but all other fields are optional due to the potential lack of information about the Lead.
- **Mail** - Mail is a collection of emails the user of Pipedrive has sent in regards to any business process or activity. It can be linked to a Person or Deal and can also be used to track emails received from a personal email account.

[9] `https://support.pipedrive.com/en/article/how-is-pipedrive-data-organized`

In the scope of this thesis, the data that is going to be the focus is limited to Organizations, People, Deals, and Activities. The reason for this is to keep the scope of the thesis manageable and the data being left out (Leads and Mail) do not have any direct relation to the feature in question, Campaigns. It is to be noted that Mail in this area of Pipedrive does not relate to the Campaigns feature, which also deals with managing emails.

3.2 Data stack in Pipedrive

Users of Pipedrive generate a lot of data on a daily basis. Whether it be from user activities or other stateful and real-time streams, it is a collection of information that can serve a lot of purposes. That being said, data has a different meaning for different stakeholders at the company.

To keep things as streamlined as possible, Pipedrive has developed a data stack, pictured with Figure 3, that caters to fulfilling requirements from the engineering department to sales, all the while having an universal base. The following chapters give an overview of how data can be processed in Pipedrive.

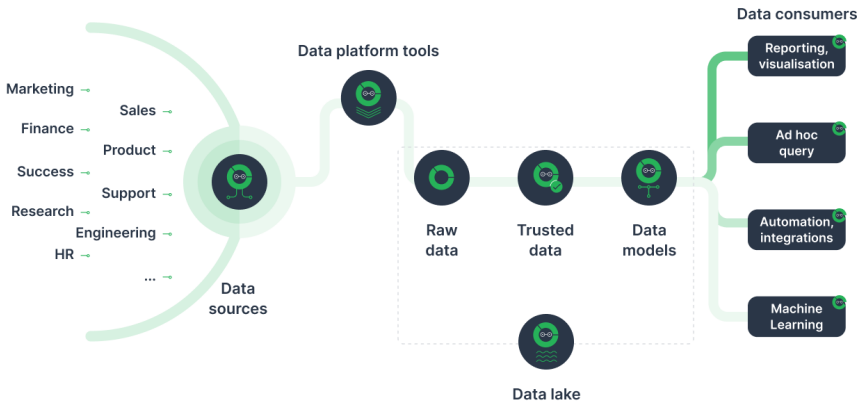


Figure 3. Data overview in Pipedrive.

3.2.1 Data Warehouse

Data Warehouse (DW) is the name given to the department who is responsible for the collection, storage, management and value extraction of the data across Pipedrive. One of the tools used by DW is the Amazon Simple Storage Service (Amazon S3)[10]. S3

[10]https://aws.amazon.com/s3/

serves multiple purposes in the context of Pipedrive. One of the biggest benefits of S3 is keeping CPU and storage separate so that they are loosely coupled, thus giving flexibility to scale either one independently. Furthermore, it is:

1. the staging area for incoming data;
2. the data source and destination for Apache Spark jobs;
3. the primary location where data is archived.

Regarding the final point, all data in DW is maintained at a granular level where specific user data is anonymized. This gives an opportunity to go back at any point in time to look at the data from a different perspective.

This also establishes the priority of keeping customers' data secure, which is handled through encrypting data at rest, using solid key management principles and access to data being on a need-to-know basis, controlled by VPN and LDAP authentication.

3.2.2 Data Steward

Data in Data Warehouse, in its most granular form, is divided into columns and then tables, which in turn are grouped under schemas using the star schema approach. In the star schema approach, the data warehouse contains a large central table, also known as the fact table, containing the bulk of the data with no redundancy, and a set of smaller attendant tables also known as dimension tables, one for each dimension[HKP12]. Due to the sheer amount of data being processed, navigating solely through Data Warehouse can be difficult. This is where Data Steward is used.

Data Steward is an in-house built web-app for browsing and editing metadata of Data Warehouse. Its features include searching from all available Data Warehouse databases, schemas and tables, with a detailed view for each table, as seen on Figure 4. The detail view enables the user to read descriptions about every column within the table, view the same table in other databases, and where the table is referenced in Zeppelin notebooks.

3.2.3 Apache Spark

Apache Spark is a fast, in-memory data processing engine with elegant and expressive development APIs to allow data workers to efficiently execute streaming, machine learning or SQL workloads that require fast iterative access to datasets[11].

It is based on Hadoop MapReduce[12]. MapReduce is a programming model and an associated implementation for processing and generating big data sets with a parallel, distributed algorithm on a cluster. A MapReduce program is composed of a map

[11] `https://spark.apache.org/`

[12] `https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html`

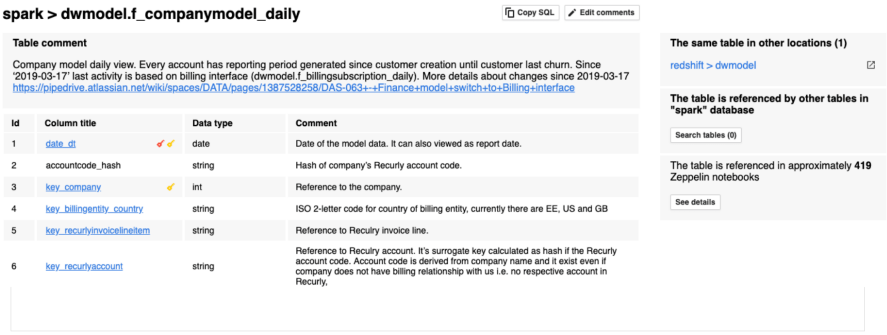


Figure 4. Example of Data Steward table detail view.

procedure, which performs filtering and sorting, and a reduce method, which performs a summary operation. Apache Spark extends the MapReduce model to efficiently use it for more types of computations, which includes interactive queries and stream processing.

Pipedrive has chosen Apache Spark as a part of its data stack for the following features:

- Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing number of read/write operations to disk and storing the intermediate processing data in memory;
- Spark supports multiple languages, providing built-in APIs in Java, Scala, or Python;
- In addition to 'map' and 'reduce', Spark also supports advanced analytics, with support to SQL queries, streaming data, machine learning, and graph algorithms.

3.2.4 Apache Zeppelin

Apache Zeppelin is an open source web-based notebook system used for enabling scenarios involving data in Pipedrive. This includes data transformation, ingestion, ad-hoc analysis, reporting, presentation and interactive collaboration.

User interaction for querying and preparing data is done via Zeppelin Notebooks. A notebook consists of paragraphs, which in turn have two sections: an editor for writing code; and the output of the execution of that code. A visual overview of paragraphs, editors, and outputs can be seen with Figure 5.

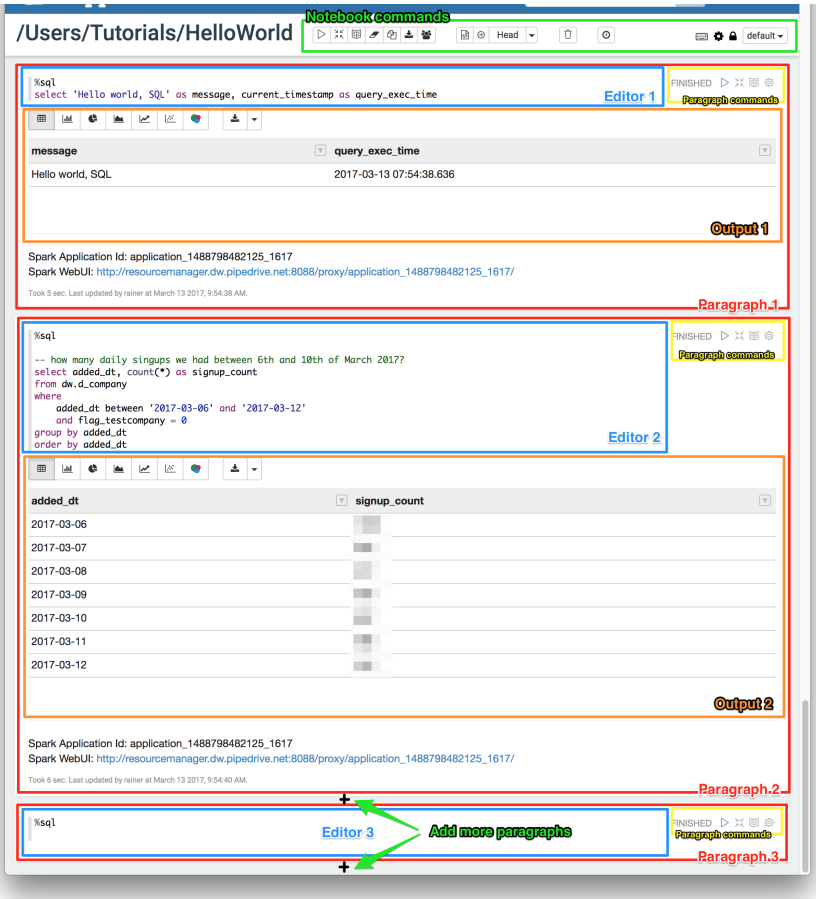


Figure 5. Example of a Zeppelin Notebook view.

3.2.5 Segment.io and PDW Sesheta

Segment.io[13] is a customer data platform that is used by Pipedrive to collect, clean, and activate their customer data through one interface. Pipedrive uses Segment for UI tracking events and low volume backend events.

PDW Sesheta is an instrumentation framework that enables collection of events and other data points in a scalable manner. Its main goal is to make instrumentations of the Pipedrive application easy and affordable. Although it is mostly obsolete in its usage being replaced by Segment.io, it is still used to collect some tracking events which are being used in this thesis.

[13] `https://segment.com/`

3.3 Data schemas

Data in Pipedrive can be split into three layers, based on its granularity and processing efforts:

- **Datalake Bronze Layer** - This is for data in its most raw form, where it is ingested and stored for usage in the Silver and Gold Layers. The schemas relevant to the thesis in this layer are `sesheta_{clientname.eventname}` and `segment_{projectname.eventname}`.

- **DW Silver Layer** - In this layer, data from the Bronze Layer is processed by automated cleaning, filtering and enrichment. The relevant schema in this layer is `dw`, the DW main schema.

  The `dw` schema tables have a convention of having a prefix assigned to them relating to the content of the data the table contains within:

  - prefix `'d_'` stands for 'dimension' and is used for describing tables containing different attributes of the entities involved in business processes, such as '`d_time`' and '`d_currency`';
  - prefix `'f_'` stands for 'fact table' and is used for describing tables that store quantitative information, for example the number of daily active users, '`f_activeuser_daily`'.

- **DW Gold Layer** - This is also known as the "Query Layer". The data in this layer is prepared for creating easy reports and business-level aggregates with DW models. In this thesis, no schemas from this layer are used, as the aggregation of data was deemed to be insufficient in granularity to create the necessary relations between data from different tables.

Having defined the relevant areas of data in Chapter 3.1 with the different layers of data in Pipedrive in this chapter, it is now possible to elaborate on the schemas of tables with relevant event logs that are queried for the preparation of the data required for training the model.

An overview of tables used in this thesis is brought out in Appendix I. The description of data extracted from each table is as follows:

- **dw.f_customerdeal** - Information about deals' relations per company, containing the deal's ID along with the contact person's or organisation's ID. In the case of no relation to a person or organisation, the ID is set to -9999.

- **dw.d_customercontactperson** - Information about the contact person with the organisation ID, if they belong to one. No deal ID is supplied as one person can be linked to many deals.

- **dw.d_customercontactorganisation** - Information about the contact organisation, containing its ID. Neither deal ID nor person ID is supplied as one organisation can be linked to many deals and many people can belong to one organisation.
- **sesheta_webapp_live.deal_won** - Information about events where a deal was marked as 'won'. Included are the deal ID along with the person or organisation ID involved with the deal.
- **sesheta_webapp_live.deal_lost** - Information about events where a deal was marked as 'lost'. Included are the deal ID along with the person or organisation ID involved with the deal.
- **sesheta_webapp_live.activity_added** - Information about events of adding an activity related to deals, contact people, or organisations. Note that although in Pipedrive there are different activity types, this thesis handles all types of activities added as the same type of event.
- **segment_sus_webapp.track_email_campaign_sent** - Information about whenever a user has sent out an email campaign via the Campaigns feature. This data is necessary to create a distinction of events into the event prefixes and event suffixes, where the event of sending out a campaign for the first time in a company is the splitting point.

Let it be noted that in Appendix I, only columns relevant to this work are described, with columns of the same name between tables indicating to the same set of data.

3.4 Criteria for suitable datasets

Chapter 3.3 elaborated on the tables where suitable event logs can be queried from. This chapter further elaborates on the conditions we set that should apply for the data in order to conduct training on the machine learning model.

1. Only event logs with a date of later than January 1 2021 are considered. This is due to the fact that the Campaigns feature was officially released to public in February 2022, with first usages in summer of 2021. In order to to maintain the balance of event prefixes and suffixes, only relatively recent event data is processed. Furthermore, data dating from before 2021 has a higher chance of being characterised by user behavior that might not apply anymore to the company in question.
2. Only companies that have an event log of having sent out an email campaign are considered. The reason for it is that for this thesis, the machine learning model is being trained to recognize the patterns between events that occurred before

enabling the Campaigns feature and events after enabling it. Without these events, it would not be possible to create event prefixes and suffixes

3. Furthermore, sufficient time has to have been passed after the first email campaign of a company was sent. This relates to the first point mentioned previously of maintaining a balance between event prefixes and suffixes. Additionally, event traces that have their splitting point earlier are preferred to later ones, as it gives an opportunity to see how sending out campaigns has affected the metrics of a given company. A sufficient time in this case is at least two months since the sending of the first email campaign, with the end of February 2022 being considered the latest possible time.

4. Of the remaining sets of events that fit the criteria, those that have at least 10 000 and less than 100 000 event logs are considered. We deemed less than 10 000 events to not be enough data to train the model and we also chose 100 000 events as an upper limit to take into account the capabilities of the hardware present to them at the time of training.

5. Finally, a constraint of event sequences that had a length of less than 2 either the prefix or suffix side were filtered out. This was a requirement for the training of the model to correctly create a mapping within an event sequence. Also, from a business perspective, event sequences that were smaller than this could only indicate to deals that were created and then concluded with no additional activities, which do not add any value in terms of enabling the Campaigns feature.

3.5 Further preprocessing

Chapters 3.3 and 3.4 have established the type of data that is available to be worked on. This chapter looks at further preprocessing techniques to convert the data ready as input to the machine learning model.

One condition that must apply for the data is that the event names have to be of numeric value. Before processing the data, the type of event was not supplied with the event log, which meant the event type had to be derived from the table name.

This is why we propose a custom mapping of event names to arbitrary unique numeric ID that increases incrementally for each event added. Table 2 shows the events used in this thesis along with the corresponding ID.

Note that sequence order of events to ID values does not imply event having prominence over any other. An example of this is with the event `activity.added` having the largest ID due to having been the last event implemented into the mapping, but would contextually make more sense to be grouped with the ID values of other `.added` events.

Another modification was needed to be implemented - every event log must belong into an event trace, a sequence of events. An suitable event sequence in the scope of this

Name of event	Corresponding ID in log
person.added	1
organization.added	2
deal.added	3
deal.won	4
deal.lost	5
activity.added	6

Table 2. Events with their mapped ID values.

thesis would be a series of events that describe some business process from beginning to end in Pipedrive. The only business process in this case, which would include all of the event logs, would be the life cycle of a deal from its creation to its end.

In order for a series of event logs to be considered under one event trace, they all have to share the same Case ID.

Luckily, in this situation, the ID of a deal can be considered also as the Case ID, as every other event type considered is only included in the data if the event log has been generated in relation to a deal. This also extends to activity.added in the scope of adding an activity to a person or organisation, as long as that contact is related to a deal. Going forward, the terms deal_id and case_id are synonymous.

Finally, having joined together all of the data from tables highlighted in Appendix I, except for segment_sus_webapp.track_email_campaign_sent, we get a set of events in the same structure as displayed with Table 3.

key_company	complete_timestamp	activity_id	person_id	org_id	case_id
7706265	2021-10-14T16:31:48	2	0	10241	23086
7706265	2021-10-14T16:31:48	1	12417	10241	23086
7706265	2021-10-14T16:31:48	3	12417	10241	23086
7706265	2021-10-14T17:21:22	6	12417	0	23086
7706265	2021-11-17T12:43:20	6	0	10241	23086
7706265	2021-11-20T11:12:39	5	12417	0	23086

Table 3. Example of preprocessed event logs.

event_name	key_company	time
email_campaign_sent	7706265	2021-11-15T11:00:05

Table 4. Example of event log for sending a campaign.

The reason for excluding segment_sus_webapp.track_email_campaign_sent from the set of event logs is due to the reason that its events serves a different purpose than the

ones extracted from other tables. The event logs in this table are not used as an input to the training of the model, but rather as an additional preprocessing tool for the rest of event logs that are being used for training.

Using the timestamp of these events, it is possible to separate any trace of events into a prefix and suffix, as long as the timestamp of the event is between the first and final event timestamp in an event trace and criteria mentioned in Chapter 3.4 are satisfied.

3.6 Example of entire event sequence

Taking the event logs from Table 3 and Table 4 as an example of a complete event trace, meaning that all event logs with `case_id 23086` are present, this chapter explains in a business process context what information can be extracted from these event logs.

The first three events are executed at the same time, which can only happen when creating a deal. This is possible when the user fills out the details of a deal with an organization and person that did not exist in the system beforehand. Since for both the contact person and organisation to be linked correctly to the deal, a new entity has to be created for both of them before creating a deal.

More specifically, an organisation has to be created before a person and a person has to be created before a deal, if the user decides to use both an organisation and a person in their deal detail. This is due to the analogous logic that applied to deal has to apply to creating person as well, when linking the person to the organisation.

After the deal has been created, an activity is promptly created that involves the related contact person of the deal, but not the organisation, displayed with row 4 in Table 3. Then there is a longer period of pause between the events in row 4 and 5, but in between them the user has also sent out their first email campaign, as seen from the only row in Table 4. Afterwards, the final two events are logged about the deal - creating another activity, this time with the organization, and finally, ending the deal with having marked the deal as 'won'. In this particular case of an event trace, we can consider the event prefix to be rows 1-4 in Table 3, the splitting point being row 1 in Table 4 and the event suffix to be rows 5-6 in Table 3.

4 Technical Implementation

This chapter covers in depth how the data is used in conjunction with the MLMME framework mentioned in Chapter 2.7 for the model training suffix prediction.

A visualisation of the framework can be seen with Figure 6. The input is a collection of event logs which are processed further by splitting them into two sets of sequence based on a chosen timestamp. Then, the sequences are used as an input for the Adversarial Predictive Model, which consists of two parts, the generator and the discriminator. The model is trained to learn the patterns of the given event logs. Finally, beam search is used with the output from the model to generate new event logs that are similar to the ones used as input.

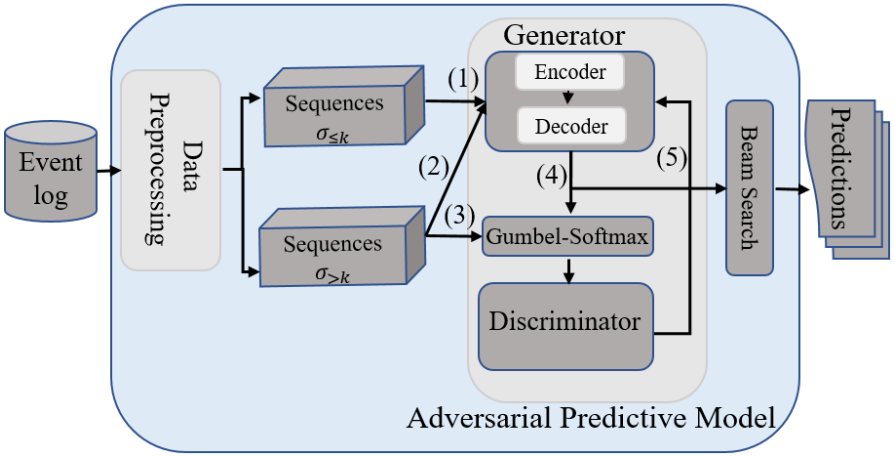


Figure 6. Framework for suffix and remaining time prediction proposed by [TLRE21].

This chapter also includes the modifications made to the framework to suit the needs of this thesis. Finally, an overview of the event suffix generation is given, interpreted in Pipedrive business terms.

4.1 Data preparation

The data extracted in Chapter 3 is served as an input in `.csv` format to the MLMME framework. A dataset consists of one Pipedrive company's event traces, including a timestamp where the dataset is intended to be split into prefixes and suffixes. Restriction to one company per dataset is done to ensure that the specific behavior of a company is captured.

The events are first encoded to a one-hot representation[HH12]. An example is highlighted with Table 5 for a single event trace. Note that the mapping of event names

to integers created in Table 2 applies as columns names in this case. Also, an additional column `'0'` has been added which indicates the end of a trace along with two columns `duration_time` and `remaining_time` for remaining time prediction.

The transformed event logs are saved as a DataFrame using the `pandas` library[14].

0	**1**	**2**	**3**	**4**	**5**	**6**	**duration_time**	**remaining_time**	**class**	**timestamp**	**CaseID**
0	0	1	0	0	0	0	0.0	86237.0	2	2021-08-30 21:58:03	16217
0	1	0	0	0	0	0	0.0	86237.0	1	2021-08-30 21:58:03	16217
0	0	0	1	0	0	0	0.00003	86236.0	3	2021-08-30 21:58:04	16217
0	0	0	0	0	0	1	0.00186	22360.0	6	2021-08-31 15:42:40	16217
1	0	0	0	0	0	0	0.00	0	0	0	16217

Table 5. Example of events in one-hot encoding.

4.2 Partitioning of events

After the events have been encoded, the event logs are partitioned into sets of logs that are of similar size. The number of partitions is determined by EQ(1), where t is a list of event traces, the Min and Max functions respectively return the longest and shortest event trace in the list and w_p is the width of partition, indicating what the difference between the longest and shortest trace in a partition should be. For all further testing in this work, w_p has been set to 2.

$$n_p(t, w_p) = \left\lceil \frac{Max(t) - Min(t)}{w_p} \right\rceil \tag{1}$$

The reason for partitioning the event traces is to more easily create event prefixes and suffices that are of similar size, which is simplified when an assumption is made that the event traces being split are of roughly the same size. This also attempts to categorize traces of similar length as traces with similar user behavior.

When partitioning is done, event prefixes and suffixes of variable length are created by iterating over partition groups. Variable prefix and suffix length is used to prepare more of data for the model to train on. The algorithm that does prefix and suffix creation is displayed with Algorithm 1.

In the beginning of the algorithm, the maximum event trace length m is computed to determine the number of sets of suffixes and prefixes that are going to be created. For every such length $i \in [2, m]$ the generated partitions is iterated over and the events in each partition $partition$ is grouped by the case IDs present.

For every group of cases in $partition$, prefixes are determined by filtering out any event that occurred after timestamp t. If the number of remaining prefixes is equal to

[14] `https://pandas.pydata.org/`

Algorithm 1: creatingPrefixSuffix

Input: List of event partitions P; Timestamp for splitting events t; Max trace length m

Result: A dictionary D of tuples that contain three items: list of tensors for event prefixes p, list of tensors for event suffixes s and a list c of case IDs related to events p and s

$D \leftarrow \{\}$;
foreach *i in* $[2, m)$ **do**
    $p, s, c \leftarrow []$;
    foreach *partition in P* **do**
        $group \leftarrow groupBy(partition, CaseID)$;
        $p_i, s_i, c_i \leftarrow []$;
        foreach *caseID, g in group* **do**
            $prefixes \leftarrow g[g[timestamp] < t]$;
            if *len(prefixes) == i* **then**
                if *len(g) > len(prefixes)* **then**
                    $append(p_i, makeTensor(g[0 : i]))$;
                    $append(s_i, makeTensor(g[i :]))$;
                    $append(c_i, caseID)$;
        if $len(p_i) > 0$ **then**
            if $len(s_i) > 0$ **then**
                $append(p, p_i)$;
                $append(s, s_i)$;
                $append(c, c_i)$;
    $D[i] = (p, s, c)$;

i and there exist at least some suffix events after timestamp t, meaning length of the group is larger than the length of prefixes, then tensors are created using the prefixes and suffixes.

Finally, if any suitable sets of prefixes and suffixes were found, then they are stored in a dictionary of tuples, where the key is i and teach key contains a tuple of three items, a list of tensors for event prefixes p, a list of tensors for event suffixes s, and a list c of case IDs related to events p and s.

Before starting the training, the preprocessed data object is saved as a `.pkl` file, a format which enables the user to save a Python object structure in a static state to a file,

using the Python module `pickle`[15]. This is useful when dealing with larger datasets, as the `.pkl` file can be used in substitution of the `.csv` file to skip the preprocessing part.

4.3 Model training

The MLMME framework uses two methods for training the model, applying both open-loop training using Maximum Likelihood Estimation using Long Short-Term Memory (LSTM) and training inspired by Generative Adversarial Nets[HS97][GPAM$^+$14].

Open-loop training is achieved by taking the one-hot encoding of an event prediction e_k, where k indicates the position of the event in an event trace e. Using e_k, it is possible to concatenate to it the predicted duration time t_{k+1}, which gives e_{k+1} This is used as an input for the prediction of the next event, creating open-loop training [TLRE21].

The adversarial training of the model involves two neural networks, the generator and the discriminator. The generator creates predictions for the discriminator, which evaluates the predictions using ground truth.

In regards to adversarial training logic, the generator first outputs a set of predicted event suffixes that is served as an input to the discriminator, along with a set of suffixes from the training set. Then, the discriminator takes these two sets and attempts to assign high probability values to the events from the set of ground truths while keeping the probability values of the predicted events low. Finally, generator takes this feedback from the discriminator through backpropagation and tries to create new logs that could make the discriminator execute the probability assignment incorrectly, through which it trains itself to generate more and more realistic event logs.

This approach creates a situation where the discriminator and generator are in opposing positions. This results in the generator receiving feedback in more ways than just regular training, also capturing the temporal attributes of a sequence of events in addition to the patterns of event suffixes.

It should be noted that the framework has an option of executing the training with or without adversarial training. The training done in the scope of this thesis pertains only to training the model with adversarial training, as it showed better overall results in previous uses, although granted, the difference was not statistically significant [TLRE21].

4.4 Suffix generation

After training has concluded, it is then possible to use the generator to start predicting event suffixes.

First, a probability distribution over all possible activities is calculated.To alleviate the greediness of always picking the single most probable option and to optimise the search space, beam search is used - a pruned version of breadth-first search [MVC20].

[15] `https://docs.python.org/3/library/pickle.html`

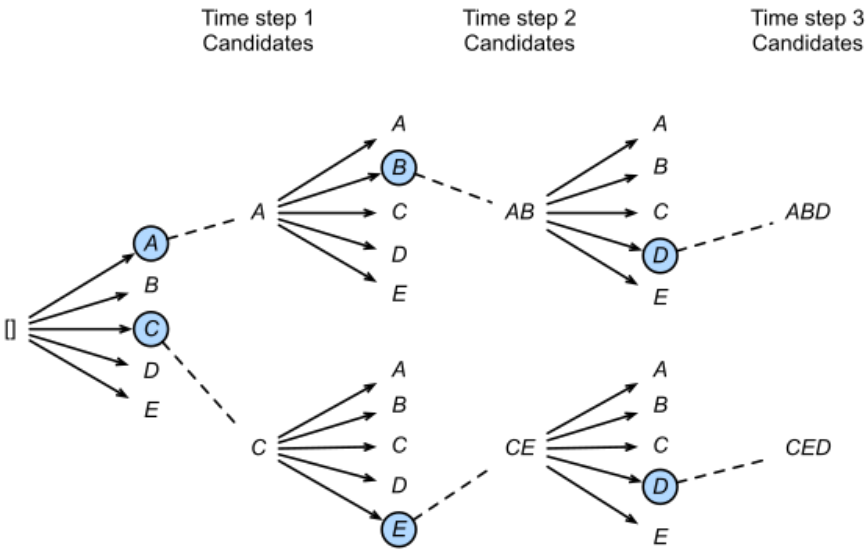


Figure 7. Visualization of beam search algorithm.[16]

A visualization of the algorithm can be seen with Figure 7. In this prediction context, beam search algorithm looks at n most probable predicted activities at any given time step, where n is a pre-determined beam width. Then, the activities that are not in the top n candidates are discarded and the prediction for the next time step can begin. The process of prediction along a selected activity path is continued until the event that indicates the end of a sequence is generated or that event is not considered in the top n candidates anymore. When no partial predictions are left to be made, beam search algorithm finishes.

The "trail" of activities that is predicted using beam search is considered to be a new event trace that mimics the user behavior of the event traces the model was trained on.

[16] https://d2l.ai/chapter_recurrent-modern/beam-search.html

5 Feedback collection

To answer the RQ2 stated in Chapter 1.4, this chapter gives an overview on how feedback about the technical solution was collected.

5.1 Approach to collecting feedback

Since the technical solution was initially intended to give an additional tool to Pipedrive employees in the marketing department, it was decided that feedback should be collected from key people well versed in that area, to get an understanding if the tool meets their business needs and collect input for possible improvements. In order to align the technical part with expectations from Pipedrive, a comprehensive overview of the solution is needed to be given to convey it effectively to stakeholders.

The best way to do this was to conduct an interview with Senior Product Marketing Manager James Campbell, who was the person that had been the primary contact person throughout the development of the thesis. Additionally, he was the person who had proposed the initial thesis topic idea to start looking at potential solutions to problems in the marketing area of Pipedrive.

5.2 Structure of interview

The interview was constructed to consist of two parts. The first part is a slide presentation conducted by the interviewer to give a high-level overview of the solution to the interviewee[17]. This includes the steps we conducted in the technical approach, along with examples of where the solution could be implemented.

The second part involved a section of Q&A with the interviewee that served the purpose of gaining answers for the feedback questions listed below. In order to receive answers that could provide more value for analysis, the questions were first discussed in a live format, after which the interviewee was asked to fill out the questionnaire by themselves. This was also done to add confidence that the questions would not be misinterpreted when they were answered independently.

5.3 Feedback questions

In order to gain insight in more ways than just a 'yes' or 'no' answer, a questionnaire was created that enables to answer RQ2 from multiple different aspects of Pipedrive [18]. Below are listed the questions that were asked to the interviewee with further elaboration

[17] https://docs.google.com/presentation/d/1NGHG-AfSJvBOLq6I9QhhMpt08ZIN2-jd4pVJjoKENt8/edit?usp=sharing

[18] https://forms.gle/Cs2m4ArMhXws7bjdA

for each question as to the reason for asking it. Note that these questions were asked of the interviewee after we had given an overview of the solution.

1. What aspects should the solution focus on to provide more value to Pipedrive?

   This was the first question that we asked after the presentation. The reason it was placed first was the rest of the questions, unlike this one, are proposed in a way that enables the interviewer and interviewee to have a discussion about the topic at hand.

   Whereas, the answers about this question specifically are something that the interviewee would inherently think about throughout the overview presentation as to the extent on their first impressions. So, to get the initial thoughts out of the way, answering this would enable the people involved in the interview to establish their expectations about the topics in the next questions.

2. With this approach, what could be the most beneficial metrics to be predicted?

   This question pertains to the scope of the thesis. As mentioned in previous chapters, one of the restrictions about the implementation of the solution was that the approach would only consider the change of win and loss rate of deals.

   It is a metric that is easy to measure due to having specific events for both a win or lose scenario and won deals is the metric that most users of Pipedrive would want to maximise, but it would help the marketing department only on a user-level for promotional purposes. To gain additional insight on what other metrics Pipedrive would gain benefit from, this question was proposed.

3. Besides add-ons, do you see any other applications for this solution?

   In this work, the model training was restricted to the frame of seeing how enabling an add-on for a Pipedrive user would change some predefined metric.

   However, the solution is scalable to splitting the data into prefixes and suffixes by not only the time when an add-on was enabled, but actually any type of event log. This, in theory, enables to train the model for various custom scenarios based on the user's won behavior. The purpose of this question was to query what might these custom situations be.

4. Overall, could the results of the proposed approach provide (promotional) value to the Pipedrive marketing department for promoting their products?

After the interviewee had answered the previous questions, the assumption was made that the interviewee had given enough thought into the topic at this point and had their questions answered by the interviewer to make a judgement on the viability and usability of the solution.

5. Any other feedback?

   This question was added as a fail-safe with the intent to make sure that the interviewee had opportunity to express any ideas that had not been suitable as an answer for any of the previous questions.

6 Results and analysis

This chapter focuses on the empirical evaluation of RQ1, including experiments done with the solution, with elaboration on the results and analysis of the feedback given to the solution.

6.1 Experimental Setup

From the data criteria that was established in Chapter 3.4, six different datasets were extracted that fit the conditions for training. Table 6 gives an overview of each dataset with their number of event logs, average length of event traces, the standard deviation of trace lengths and the maximum length of traces in that dataset.

Note that the names of datasets are arbitrarily chosen and can be interpreted as the datasets of six different Pipedrive user companies.

Dataset name	Number of event logs	Avg length of traces	Std of trace lengths	Nr of traces	Max length of traces
Company 1	13542	2.147	0.798	7723	11
Company 2	18447	2.824	1.045	9412	12
Company 3	26699	3.066	1.102	13061	17
Company 4	48800	5.458	3.267	11573	42
Company 5	53276	5.965	3.464	14514	59
Company 6	65695	6.156	4.168	15264	67

Table 6. Statistics of company datasets used in experiments.

The code of the created solution is uploaded to a Github repository[19]. For each dataset in Table 6, the code was run using CUDA[20] on a Windows 10 machine with the following specifications:

GPU: NVIDIA GeForce RTX 3080

Memory: 16384MB RAM

Processor: Intel i7-10700KF CPU @ 3.80GHz (16 CPUs)

Furthermore, regarding the model, the model was set to train for 1000 epochs for each dataset, with an additional end condition being that early stopping is initiated when the accuracy on the validation set has not changed in the last 30 epochs.

[19] https://github.com/aloaasmae/mlmme-pipedrive

[20] https://developer.nvidia.com/cuda-toolkit

6.2 Evaluation Metrics

Before taking a look at the results, some evaluation metrics need to be established. First, to be able to compare similarity between event traces, Damerau-Levenshtein (DL) distance metric is used [Dam64].

Given two event sequences s_1 and s_2, where $len(s)$ is the number of events in s and a DL distance between s_1 and s_2, the similarity between the two event sequences can be considered as SDL, given in Eq. 2.

$$SDL(s_1, s_2) = 1 - \frac{DL(s_1, s_2)}{Max(len(s_1, len(s_2))} \tag{2}$$

$SDL \in [0, 1]$, where 1.0 refers to identical event sequences and 0.0 to entirely distinct sequences.

Secondly, to measure the predicted remaining time of an event trace suffix, Mean Absolute Error (MAE) is computed using the absolute error between the predicted and ground truth event traces. For this metric, lower values hint at increased similarity, with 0 indicating no error between prediction and ground truth. In the scope of this work, MAE is used only for reporting the temporal dynamics of event logs, but is not the focus of improvement. That being said, a nice-to-have goal of achieving a median MAE of less than 0.1 on at least one of the datasets was also established and all of the datasets having a median MAE of less than 1 day.

6.3 Results and Analysis

For each company dataset, beam searches of different beam sizes are conducted, as seen on Figures 8 and 9. A total of 10 beam searches were done on each company, consisting of different beam sizes in a range of $[2, 20]$ with an interval of 2 between each consecutive beam size. An upper limit of 20 is set as it appeared to be the beam size for which the suffixes could be generated in under a minute.

From Figure 8, we can see that for all datasets, as the beam size increases, the SDL score has also a trend of increasing in a logarithmic way. This is most likely due to the nature of the beam search itself, as it attempts to mitigate the greediness of best-first search by considering multiple best options. As the beam size increases, the chance of having already found the best option increases, which in turn means that finding further improvements has diminishing returns.

As for the results of median MAE of remaining time, seen on Figure 9, we see that every dataset except Company 4 has a decreasing trend, meaning the Mean Average Error is getting smaller over time. However, the trend is noticeably more gradual than compared to the trend of SDL scores from Figure 8. This may hint that beam search already finds the best option with already a small beam size, meaning that the a greedy approach works well here, or the difference between the next best candidates is small

enough that the choice does not have a major impact on the result even if beam search had a wider selection to choose from.

The best results against test set validation of training the model on a per dataset basis can be seen in Table 7.

Dataset name	Average of SDL	Median of MAE (in days)	Beam search beam size
Company 1	0.8092	0.5545	20
Company 2	0.8258	0.4371	20
Company 3	0.8711	0.4186	20
Company 4	0.9038	0.2403	20
Company 5	0.8806	0.2937	20
Company 6	0.9014	0.0773	20

Table 7. Statistics of company datasets used in experiments.

From these results, some observations can be made. The average of SDL has a trend of being higher the larger the dataset is along with the maximum length of traces. The best results were provided by the dataset of Company 6, which had the largest number of event logs.

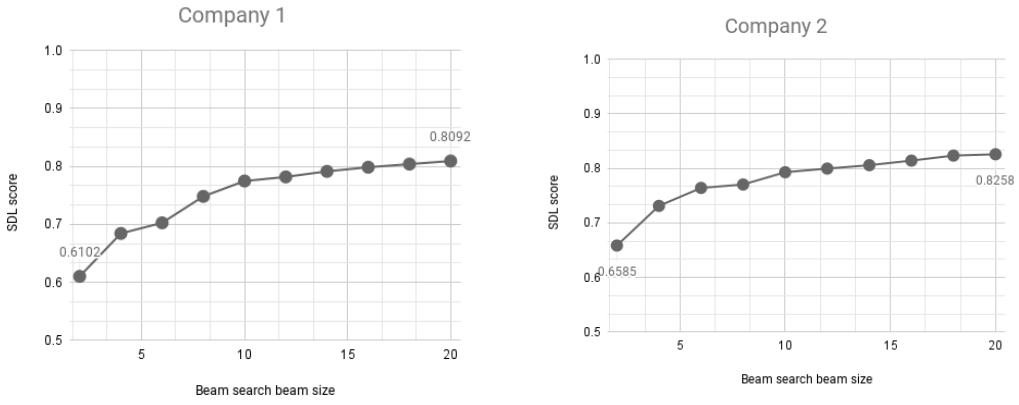
The increase in standard deviation of event trace lengths does not appear to have an effect on the overall SDL score. This may be attributed to the fact that datasets with a larger number of events inherently brings along an increased number of traces, which in a business context correlates to more cases of Deals in a company.

From the side of remaining time prediction, The MAE of every dataset for predicted times stayed under 1 day, with the best result being less than 0.1 (0.0773), which amounts to 111.3 minutes. With such MAE, other potential use cases could be solved with the same model, for example predicting the next action of a user. In a business context, these results are acceptable as the time error would still be within one working day and aligns with predicting long-term metrics over real-time ones.

Overall, having achieved an SDL score of at least 0.8 on all datasets gives confidence to the end-user stakeholders for that the inferred results are close to reality and that the approach is viable in a business context such as Pipedrive.

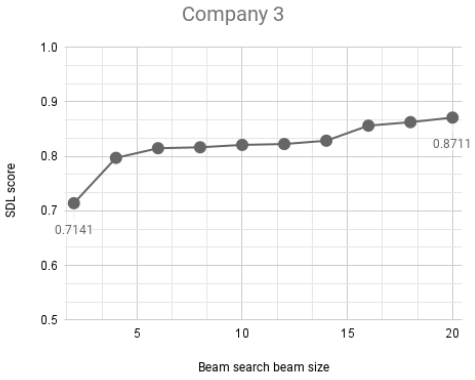
6.4 Feedback Analysis

To answer the second research question, "Do the results of the proposed approach provide (promotional) value to Pipedrive marketing department for promoting their products?", this chapter takes a look at the feedback about the solution collected from the people of the marketing department at Pipedrive.

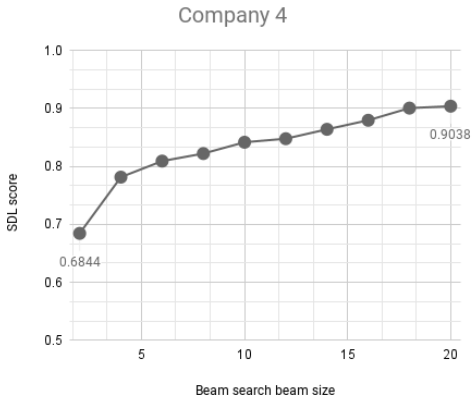


(a)

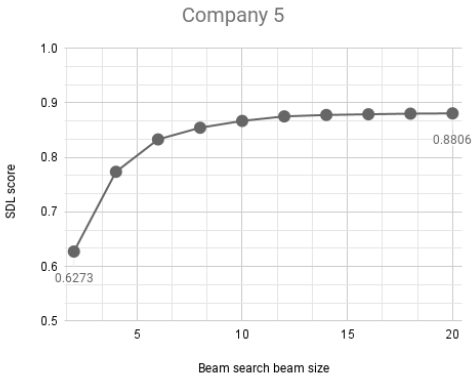
(b)



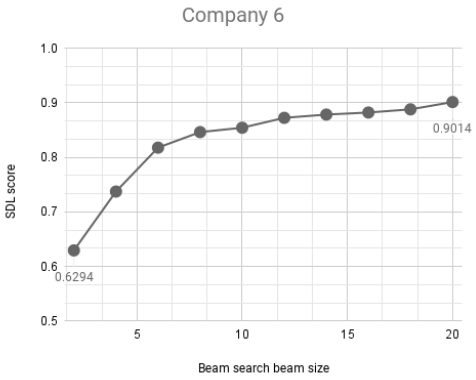
(c)



(d)



(e)



(f)

Figure 8. SDL scores for datasets across different beam sizes.

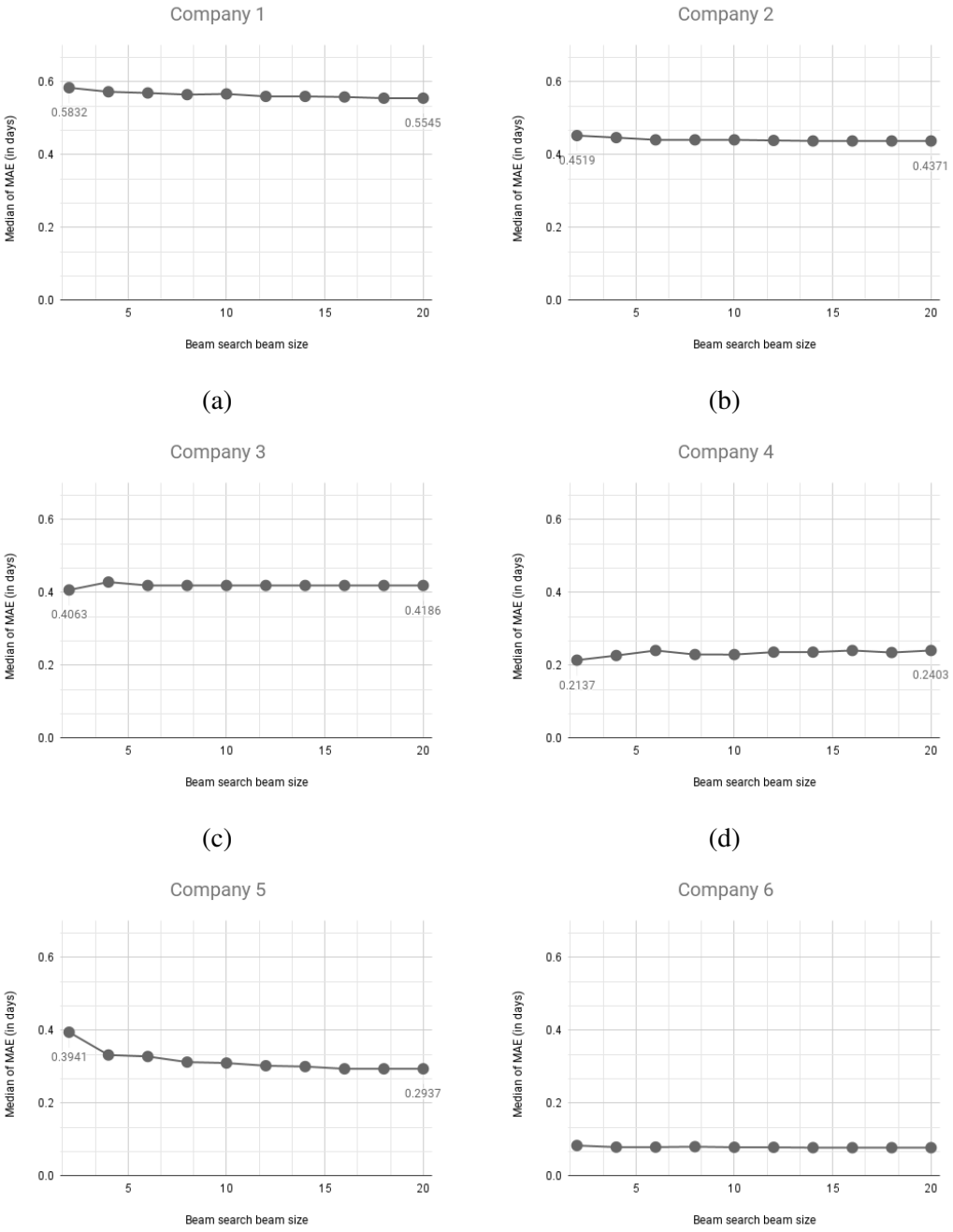


Figure 9. Median MAE of remaining time for datasets across different beam sizes.

The interview conducted with James Campbell, Senior Product Marketing Manager at Pipedrive, resulted in a thorough responses to the questions established in Chapter 5.3. Sent as a collective single answer on behalf of the marketing department, James took the initiative of talking with additional key people after the interview to obtain their perspective on the topic before submitting the responses through a form.

The responses sent are as follows, grouped by question with a chapter for discussion under each question. Note that some of the responses are edited to protect the privacy and confidentiality of Pipedrive.

What aspects should the solution focus on to provide more value to Pipedrive?

> **Response:** "Being able to use this system to create a demo account for anyone to be able to show and tell or record Pipedrive in a working-state as its meant to be used.
>
> Many teams outside of Marketing as well as within are wasting time creating accounts that are poor representations of what Pipedrive can do in a working environment, and having such a system for multiple industries and personas would speed up content production, internal enablement, support, customer success outreach, partner enablement, influencers, the list goes on.
>
> Additionally knowing that we would be able to use this to determine the most desired actions for key personas."
>
> **Discussion:** There are two insights given with this response.
>
> The first refers to use the trained model and its predictions to create internally usable accounts for external demoing purposes. While this was not the purpose the model was created for, this goal is achievable by creating a reverse mapping from the events that are predicted by the model to some API calls that execute the predicted event into a newly created account. The condition that should be fulfilled here is that in order to do this, some pre-existing data must already be present on which to predict new data on. On the other hand, the benefit of this would be tailor-made dummy accounts with very specific behavioral data that would be difficult to replicate by hand over a longer period of time.
>
> The second insight given is determining next actions for users of Pipedrive. This aligns with the initial intent of the trained model, with the key output being increased value in Product Marketing Messaging. This would enable the marketing department to send out more valuable offers to customers with an increased probability of success being backed up by the predictions of the model.

With this approach, what could be the most beneficial metrics to be predicted?

Response:

- "What were the key actions that our longest retained customers took over time?
- This model should be used to predict the actions of our key personas
- What is the workflow of a successful ideal user? And can we improve that workflow with better onboarding and make their lives even easier while scaling their success to others like them?
- Won deals
- Added users
- How many deals a persona really needs (keeping data limits in mind)
- Add-on trial adoption
- Add-on paid conversion"

Discussion: There are multiple aspects to touch upon here.

If the first metric "What were the key actions that our longest retained customers took over time?" could be determined by the model, then it would give great insight into what what separates long-term users of Pipedrive from short-term ones. As there appears to be a correlation between the retention of a user and the value Pipedrive can offer to them, defining the key actions of longest retained customers would be beneficial to paving the way for even more of these types of users.

The metrics such as predicting the ideal user falls into the similar vein as the longest retained customers - predicting what sort of potential value can be generated to the user as soon as possible.

The second aspect that is highlighted is more related to the usage of features in Pipedrive. Having the ability to predict whether a user would convert from a trial user to a paying one through the add-ons they use during trial would again give the marketing department an opportunity to focus their resources in a way where both the user and people Pipedrive could benefit from. The user can receive information about what next steps would be most valuable for them and Pipedrive people can shift their focus away from offers that are predicted to have little to no impact.

Overall, two main fields of metrics emerge from this feedback: metrics related to predicting the characteristics of the user and metrics related to predicting usages of features in Pipedrive.

Besides add-ons, do you see any other applications for this solution?

"I have basically answered this within the previous questions as well. But summarized:

1. Being able to have anyone in Pipedrive or any partner be able to demo Pipedrive in various industries and personas with no admin work needed to set up an account
2. Predicting what makes customers take the actions that allow Pipedrive to continually provide value to them
3. Continually use this to improve Product Marketing Strategy and Messaging, Improve Onboarding, and possibly even UI (User Interface)"

Response: Most of the feedback here was already covered with the previous question. However, two more things can be highlighted.

The first is improving onboarding. What is meant by this is that teaching a new user how to use Pipedrive or CRM systems overall could be improved with the predictive model by applying it to users who had a known behavior of discontinuing to use Pipedrive or using it not to its full potential. Learning the behavior of these types of users, insight could be gained on what aspects of Pipedrive are difficult for a new user to understand.

The second aspect is expanding this to user interface related events. Looking at only the web application usage related events, such as navigating to a specific section or clicking on a button, different parts of the user experience could be mapped on their frequency of usage and finding relations on what layouts affect the business processes for that user, enabling different metrics for A/B testing.

Overall, could the results of the proposed approach provide (promotional) value to the Pipedrive marketing department for promoting their products?

"Very much so - it would also help Support, Customer Success, Community, Knowledge Base, Webinars, Partners, Employer Branding, and Internal Training if we can apply this to creating dummy-data to Pipedrive accounts"

Response: This is positive confirmation that there is viability for usage to the proposed solution and further potential outside the framework this initial predictive model was designed in.

Any other feedback?

"I cannot stress enough the important of being able to align with the teams creating dummy data so we can enrich our personas and then have this easy to use 'demo' system"

Response: As this is mainly a reiteration of previous responses, no additional insight was gained from this comment.

To conclude, feedback received from the marketing department at Pipedrive has been positive. There is potential even outside the initial proposed framework in using the predicted data for creating a lot of different "what-if?" scenarios for departments other than marketing.

7 Conclusion

This work done in this thesis took a look at the possibilities of applying deep adversarial model training using Pipedrive user logs. One of the outputs of this is training the model in order to predict event suffixes, given an existing event prefix.

Based on this, two research questions were proposed:

> **RQ1**: Given an event log from the CRM system of Pipedrive, can we use the deep adversarial model to predict the benefit of adding a feature to the user's account?
>
> **RQ2**: Do the results of the proposed approach provide (promotional) value to Pipedrive marketing department for promoting their products?

To answer RQ1, event logs were collected of users that had used a new email marketing feature by Pipedrive called Campaigns. The event logs were split into a set of event prefixes and suffixes based on the timestamp when a user has first started using this new feature. By creating a mapping from events prefixes to their suffixes, adversarial model training could be undertaken, using an existing framework as a basis for development [TLRE21].

After learning the patterns of what types of prefixes create which suffixes, event logs from users, who had not previously enabled Campaigns for their company, could be used as an input for the trained model to predict a "what-if" scenario of events as if the user had enabled the feature.

The results for RQ1 show that deep adversarial model training is viable by training the model over six different companies' datasets that contained at least 10 000 event logs. For all datasets, at least 80% of predicted event traces align with the ground truth, with best dataset results reaching over 90%. The mean average error (MAE) of remaining time for predicted event logs was less than 1 day, with the best model having a MAE of less than two hours.

For RQ2, feedback was collected from key people at the marketing department of Pipedrive through an interview. From the responses, it could be concluded that the results of the proposed approach does provide value to not only the marketing department, but for other departments of Pipedrive as well, such as Support, Customer Success, and others. The solution for RQ1 was received well with potential for further development outside the initially proposed framework of this thesis.

Future work of this topic in the scope of Pipedrive involves generalising the approach to event logs other than those which are related to Campaigns in order to measure custom metrics and create even more "what-if" scenarios. Furthermore, looking into a way of using the predicted event logs in order to create dummy accounts for internal demoing purposes would be beneficial for many customer-facing employees at Pipedrive.

References

[Aut21] Brandon Christopher Autrey. *Customer Journey Analysis at Pipedrive: A Process Mining Approach*. PhD thesis, University of Tartu, 2021.

[BS11] Andrea Burattin and Alessandro Sperduti. PLG: A Framework for the Generation of Business Process Models and Their Execution Logs. In Michael zur Muehlen and Jianwen Su, editors, *Business Process Management Workshops*, pages 214–219, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[CDR19] Manuel Camargo, Marlon Dumas, and Oscar González Rojas. Simod: A Tool for Automated Discovery of Business Process Simulation Models. In Benoît Depaire, Johannes De Smedt, Marlon Dumas, Dirk Fahland, Akhil Kumar, Henrik Leopold, Manfred Reichert, Stefanie Rinderle-Ma, Stefan Schulte, Stefan Seidel, and Wil M. P. van der Aalst, editors, *BPM (PhD/Demos)*, volume 2420 of *CEUR Workshop Proceedings*, pages 139–143. CEUR-WS.org, 2019.

[CM15] Claudio Di Ciccio and Massimo Mecella. On the Discovery of Declarative Control Flows for Artful Processes. *ACM Trans. Manage. Inf. Syst.*, 5(4), January 2015. Place: New York, NY, USA Publisher: Association for Computing Machinery.

[Dam64] Fred J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of The Acm*, 7(3):171–176, March 1964. Number of pages: 6 Place: New York, NY, USA Publisher: Association for Computing Machinery tex.issue_date: March 1964.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in neural information processing systems*, volume 27. Curran Associates, Inc., 2014.

[Han20] Sten Hankewitz. Estonian-founded Pipedrive sells majority to an investment firm, becomes a unicorn, November 2020.

[HH12] Sarah Harris and David Harris. *Digital Design and Computer Architecture*. Morgan Kaufmann; 2nd edition (August 7, 2012), 2012.

[HKP12] Jiawei Han, Micheline Kamber, and Jian Pei. Data mining concepts and techniques, third edition, 2012.

[HL10] Kees M. Van Hee and Zheng Liu. Z.: Generating Benchmarks by Random Stepwise Refinement of Petri Nets. In *In: Proceedings of workshop APNOC/SUMo*, 2010.

[HM16] Ijlal Hussain and Fabrizio Maria Maggi. *Ijlal Hussain Generating Synthetic Event Logs based on Multi-perspective Business Rules*. PhD thesis, University of Tartu, 2016.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. Publisher: MIT Press.

[MVC20] Clara Meister, Tim Vieira, and Ryan Cotterell. Best-first beam search, 2020. tex.copyright: arXiv.org perpetual, non-exclusive license.

[TLRE21] Farbod Taymouri, Marcello La Rosa, and Sarah M. Erfani. A Deep Adversarial Model for Suffix and Remaining Time Prediction of Event Sequences, 2021.

[vdAW05] Wil MP van der Aalst and AJMM Weijters. Process mining., 2005.

[vELLvdA15] Maikel L. van Eck, Xixi Lu, Sander J. J. Leemans, and Wil M. P. van der Aalst. PM^{2}: A Process Mining Project Methodology. In Jelena Zdravkovic, Marite Kirikova, and Paul Johannesson, editors, *Advanced Information Systems Engineering*, pages 297–313, Cham, 2015. Springer International Publishing.

Appendix

I Table schemas

I.1 dw.f_customerdeal

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Deal added timestamp
event_name	string	Name of event, 'deal.added'
person_id	int	ID of contact person
org_id	int	ID of contact organisation
deal_id	int	ID of the deal

I.2 dw.d_customercontactperson

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Person contact added timestamp
event_name	string	Name of event, 'person.added'
person_id	int	ID of contact person
org_id	int	ID of organisation that contact belongs to

I.3 dw.d_customercontactorganisation

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Organisation added timestamp
event_name	string	Name of event, 'org.added'
person_id	int	ID of contact person
org_id	int	ID of contact organisation

I.4 sesheta_webapp_live.deal_won

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Timestamp of deal won
event_name	string	Name of event, 'deal.won'
person_id	int	ID of person related to deal
org_id	int	ID of contact organisation related to deal
deal_id	int	ID of the deal

I.5 sesheta_webapp_live.deal_lost

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Timestamp of deal lost
event_name	string	Name of event, 'deal.lost'
person_id	int	ID of person related to deal
org_id	int	ID of contact organisation related to deal
deal_id	int	ID of the deal

I.6 sesheta_webapp_live.activity_added

Column title	Data type	Description
key_company	int	Company reference
added_date	timestamp	Activity added timestamp
event_name	string	Name of event, 'activity.added'
person_id	int	ID of person related to deal
org_id	int	ID of contact organisation related to deal
deal_id	int	ID of the deal

I.7 segment_sus_webapp.track_email_campaign_sent

Column title	Data type	Description
key_company	int	Company reference
time	timestamp	Activity added timestamp
event_name	string	Name of event, 'email.campaign.sent'

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Alo Aasmäe**,
(author's name)

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Benefit prediction of buying Customer Relationship Management features of Pipedrive,

   supervised by Gamal Elkoumy and Roland Kriibi.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Alo Aasmäe
17/05/2022