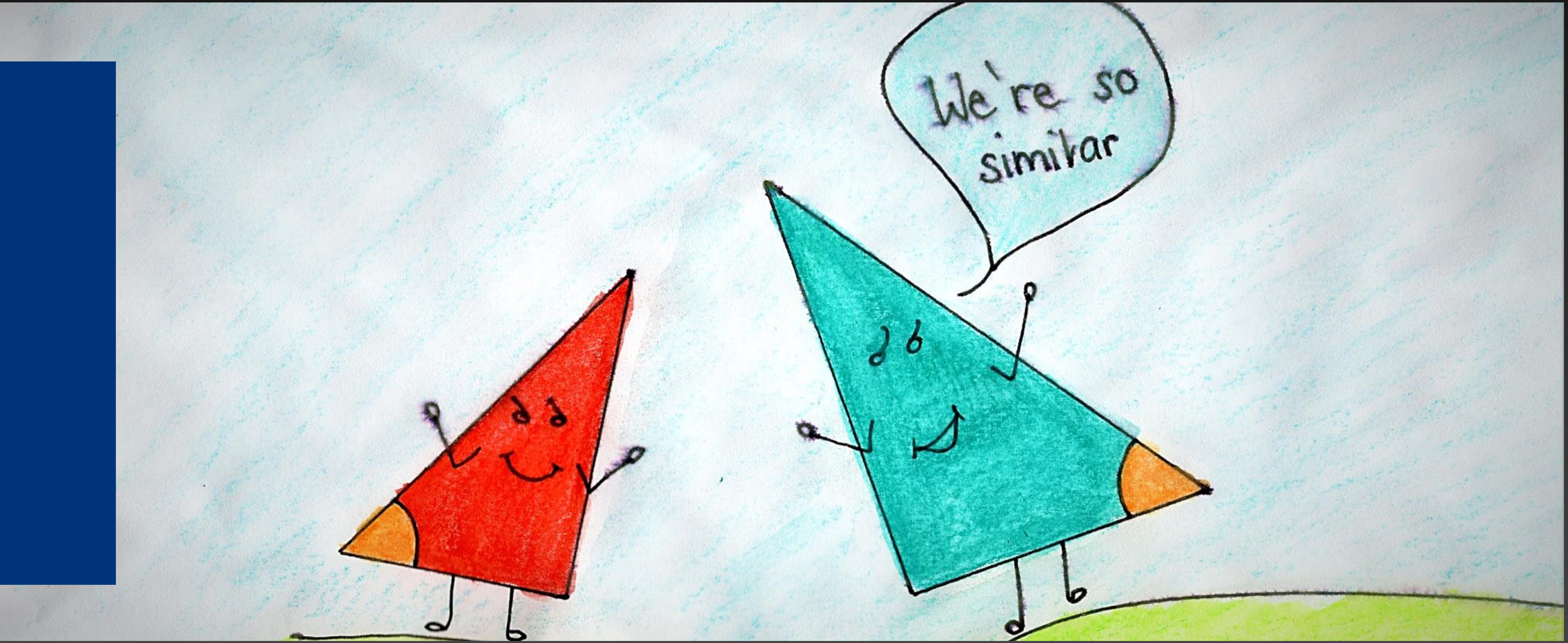
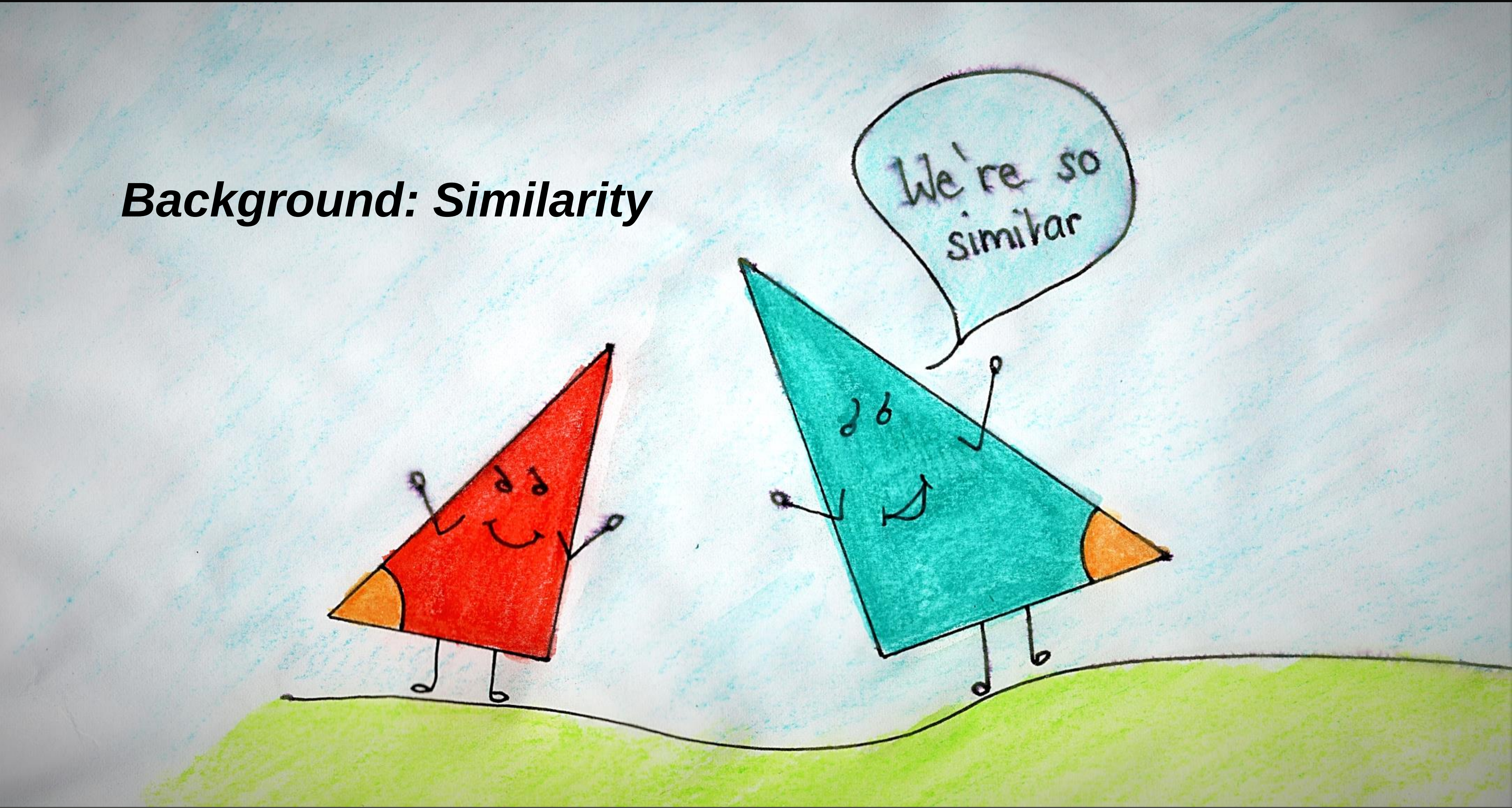
Learning and Engineering Similarity Functions for Business Recommenders

Hans Friedrich Witschel and Andreas Martin





Background: Similarity

Learning scoring functions through feedback:

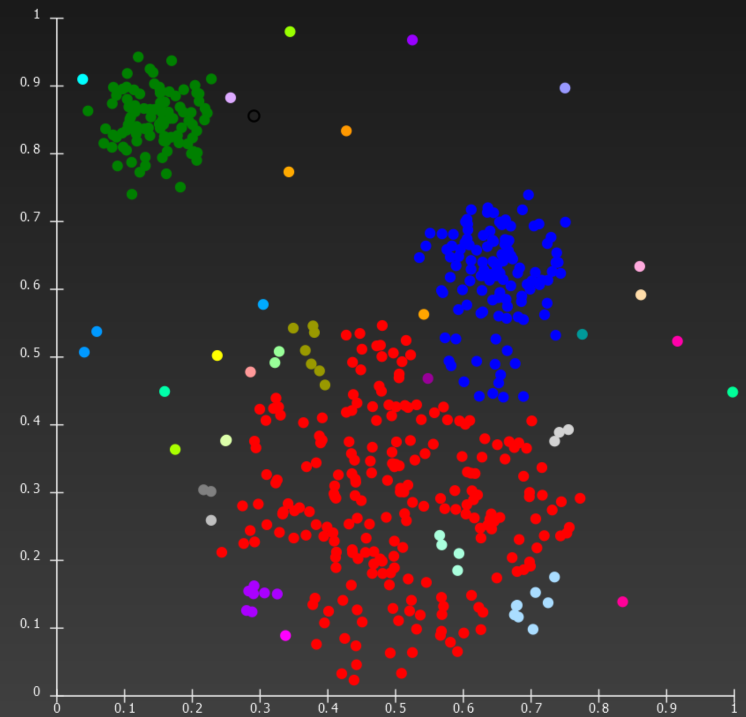
- Often, (similarity) functions are used to score items
- Functions can be made better by collecting (implicit or explicit) user feedback about top-scored items. Examples:
  - Information retrieval
  - Case-based reasoning
  - Recommender systems

Problem: Invisible Scoring (1)

Clustering

Problem: Invisible Scoring (1)

- What if scores are not what the user sees?
- Example: clustering



→ Problem: user sees clusters, but not similarities of elements!

Problem: Invisible Scoring (2)

Business Recommenders

Business Recommenders are Different

- The utility of recommendations is **defined by business requirements**, …
- … and not by a person's preferences or taste.

- They are typically **invoked much more rarely** …
- … **than recommenders that consumers** use to find products (books, music, movies etc.) of their taste.

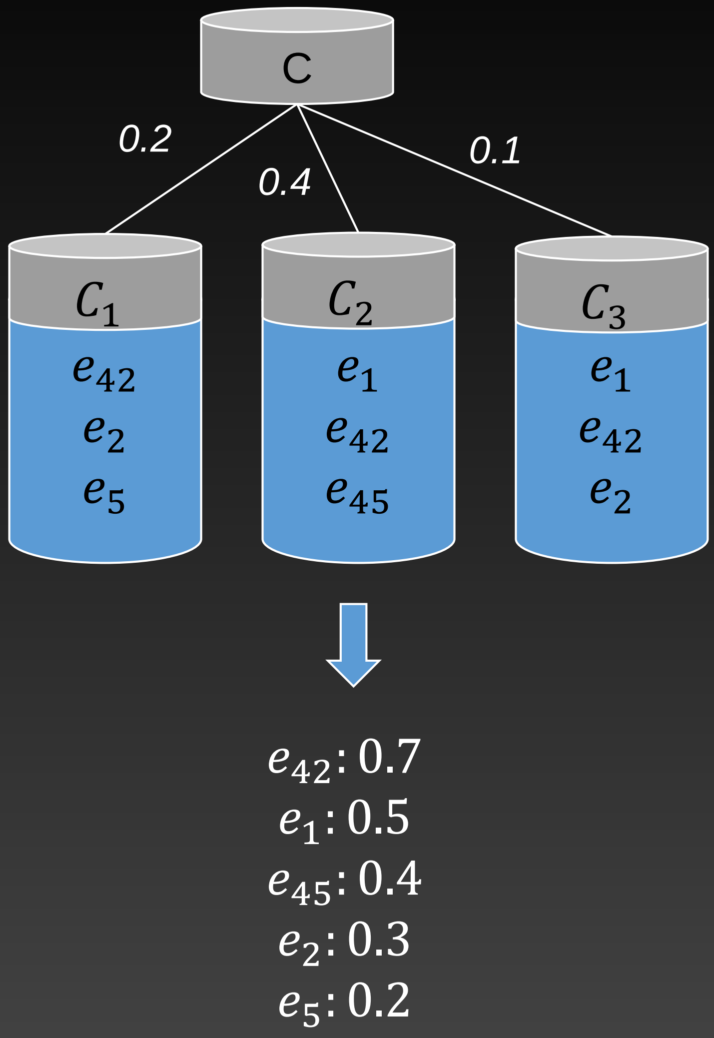
- Business recommender **have less of information** (no profile) **about a user**.
- Users need to **describe their context and requirements** in the form of a query when accessing the recommender.
- Finally, the collection of requirements or context variables can be rather complex, going beyond simple key-value pairs.

Implication

- We **can not use** the popular filtering approaches:
  - Collaborative filtering, which relies on a large numbers of user ratings and assumes that items do not have an internal structure.
    → We do not have a rating.
  - And content-based filtering, which constructs user profiles from a repeated interaction between user and system. → And we do not have this neither.

- Instead, ***case-based recommenders*** have been proposed (Bridge et al., 2005), …
- … which proceed by **constructing a description** of the business problem at hand, …
- … **retrieve** cases with a similar problem description …
- … and **combine** elements of their solution.

Problem: Invisible Scoring (2)



- In business recommenders
  - Input: a business context C
  - Desired output: a complex solution, comprising potentially many elements e_i
- Approach for independent solution elements:
  - CBR retrieval: find cases with business context similar to C
  - Retrieve all their solution elements
  - Similarity-weighted voting: rank solution elements by similarity-weighted number of cases containing them

→ Problem: user sees solution elements, not similarity of cases

Motivation (1)

Clustering and case-based (business) recommenders have thus one thing in common:

- … the result of the similarity computation is not directly visible to the user.

There is something in between the similarity computation and what the user sees

- … in clustering, it is the clustering algorithm,
- … in case-based recommenders, it is the step that selects and combines solution elements from previous similar cases.

Motivation (2) - Why is this problematic?

- In many areas where functions are used to score or rank items, such as, e.g. information retrieval, …
- … users can see and rate the output of applying the function directly.
- This allows to learn good functions by training learning algorithms with the (implicit) feedback of users (Li, 2011; Stahl, 2001; Lamontagne and Guyard, 2014).

- In clustering or case-based (business) recommenders, this is not possible …
- … since the similarity computation is (partly or fully) hidden from the user.

A Novel Similarity Engineering Process

Focus: CBR-based business recommenders

Central Insights and Assumptions

- The similarity model configuration and engineering itself is not cognitively adequate:
  - It is a cognitively intensive task for humans to build an initial case characterisation in case-based recommenders, because, firstly, **cases can have a complex structure** and secondly, the **characterisation needs to be generalized** (Martin, 2016).
  - It is a **challenging task for humans to derive from the individual mental similarity models a unified similarity model**, which can be used for a configuration of a case-based recommender.
- Humans are not good at estimating weights:
  - E.g. for weighted-sum global similarity functions. Having them do so forces them to make subjective decisions that can hardly be justified by any concrete experience or explicit knowledge (Stahl 2002).

Central Insights and Assumptions

- The similarity computation is not directly visible to humans:
  - The utility of the results that the user does see (recommendations and clusters) depends also on other algorithmic components.
  - This makes it impossible to use the feedback of humans regarding the utility of these results directly for the tuning of the similarity measure.
- Algorithms can learn weights, but rarely suggest new attributes:
  - It is hard to design them to identify and suggest missing attributes, i.e. attributes that should be additionally incorporated into a similarity measure.
  - Bad news: This is typically still a human task.
- Good news: Humans are assumed to be capable of providing feedback regarding either relative comparisons.

Related Work: Project Planning

Concrete Example: Effort Estimation

Related Work: Problem - Project planning

Situation: A company delivers **customised solutions** (software, consultancy, design,…) to their customers. A fixed price is agreed before the project starts.

Conflict:

- Either:

Effort underestimated…

Or:

Effort overestimated…

- Goal: avoid both situations by accurate estimates

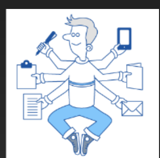
Related Work: Application Scenario and Data Set

- Use case derived from the master thesis of Von Rohr, C. R. (2017) and an advertising agency (Agentur Frontal AG).
- A comprehensive data set including over 13’000 conducted projects from the past 8 years.
- Finally, 7’946 projects remained after pre-processing (outlier removal, feature creation, text processing) for the feature selection and usage for the recommendation system.

Von Rohr, C. R., Witschel, H. F. H. F., & Martin, A. (2018). Training and Re-using Human Experience: A Recommender for More Accurate Cost Estimates in Project Planning.

Related Work: Idea - better estimation can be learned

- Intuition: there must be some patterns that can help to predict effort and that can be learned from a series of projects (→ experience)



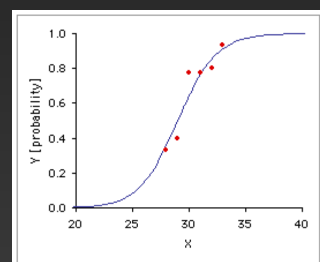
task type

customer type

?

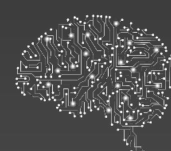
…

can it be learned?

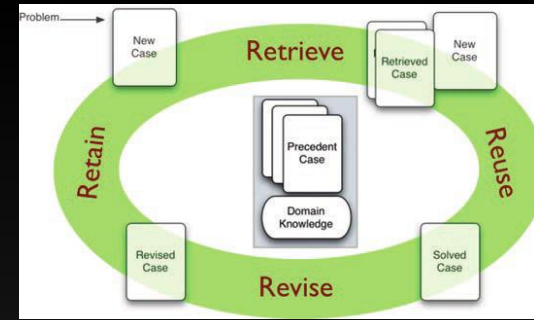


	220.00 €
	385.00 €
	754.00 €
Total	**8'624.00 €**

human



machine



Effort Estimation Process

- Observed effort estimation process (color shade: frequency of step being used):

Get informed

- Study briefing
- Identify required services
- Work package definition

Retrieve

- Search for drafts
- Search projects of same client
- Search similar projects by title
- Ask other project leads

Similarity-based retrieval

Reuse

- Copy from draft
- Copy from similar project
- Start empty project

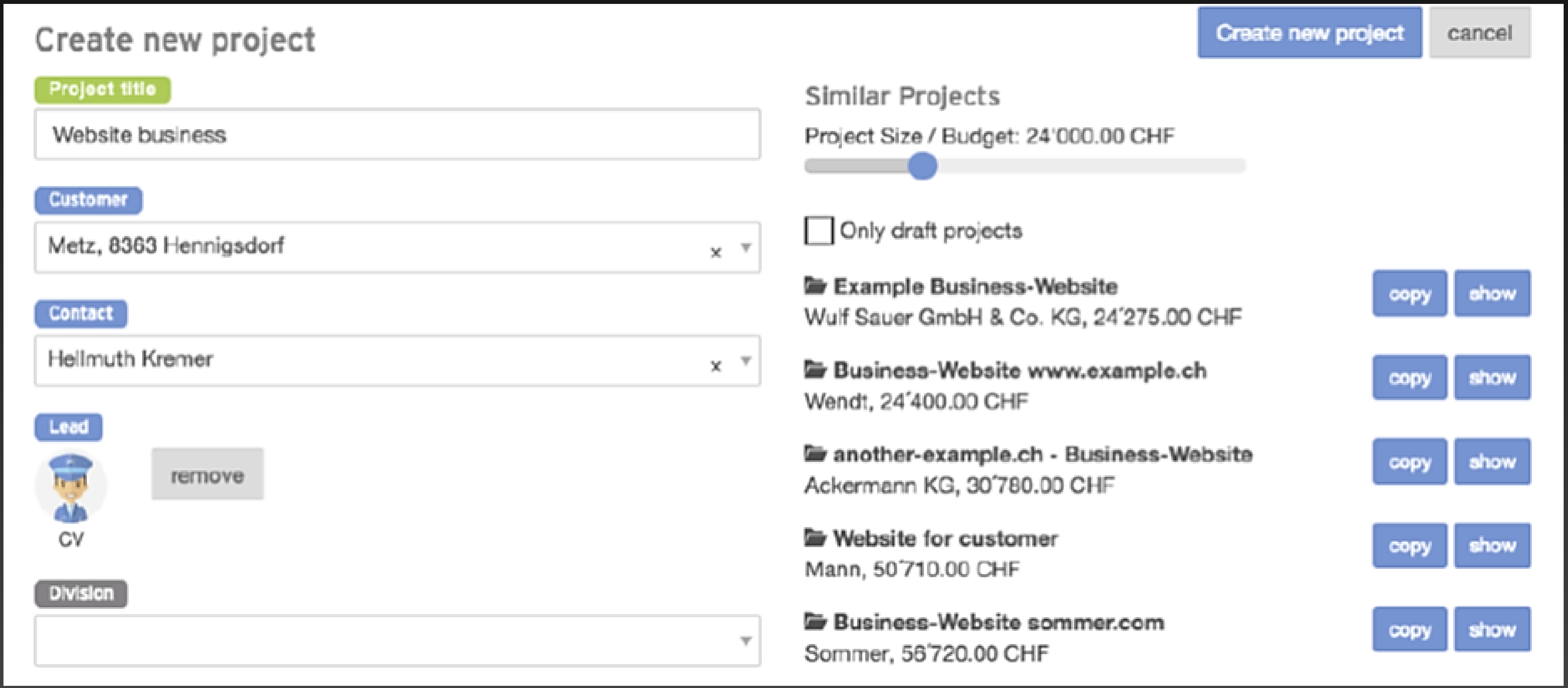
Revise

- Add positions based on experience
- Remove non-required positions
- Adjust effort based positions affected by other positions
- Adjust effort by strategy
- Adjust by customer
- Ask other project lead
- …

Recommendations

Related Work - Step 1: Retrieve Similar Projects

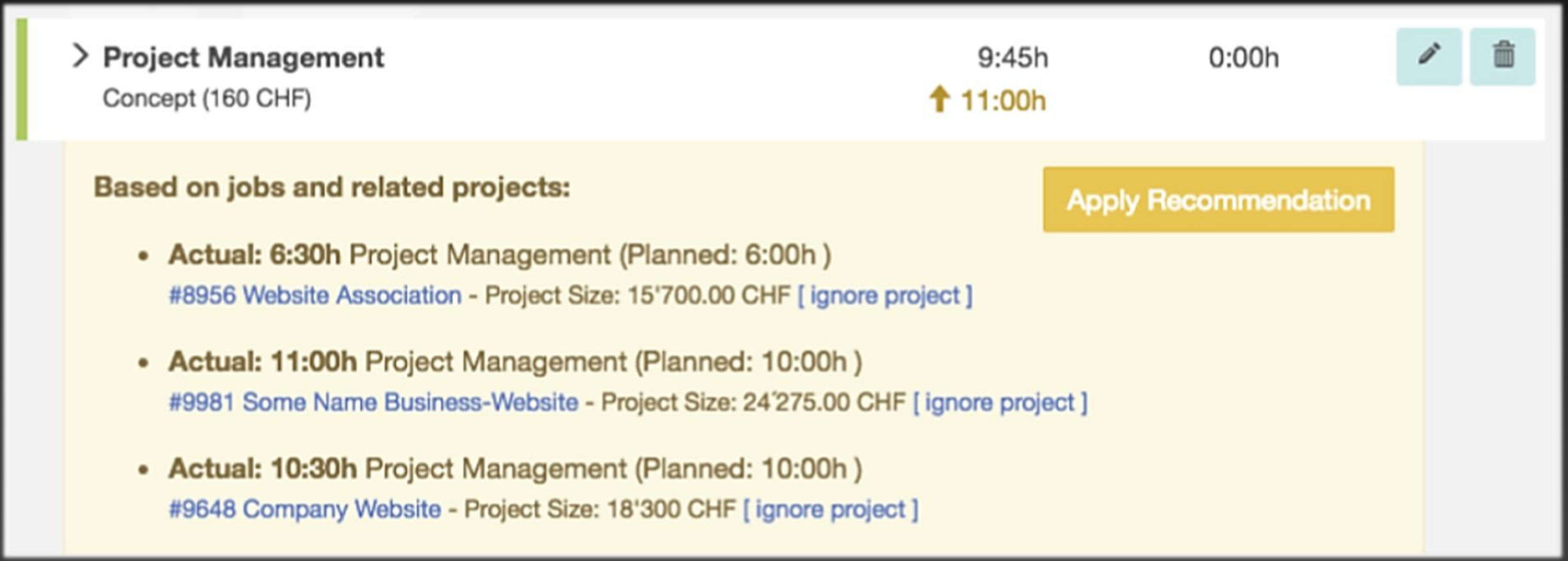
Step 1: Retrieve Similar Projects



Related Work - Step 2: Adapt and Adjust Effort

Step 2: Adapt and Adjust Effort (1)

- Approach: show similar projects to the user; allow to deselect irrelevant jobs for the current task.



- Outcome: better estimates AND: more trust of users in the result!

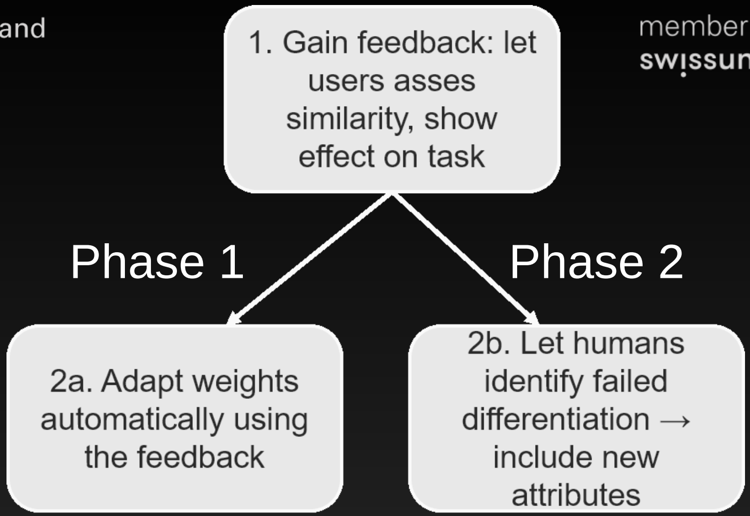
Step 2: Adapt and Adjust Effort (2)

How does it work?

- Map jobs of similar projects to this project.
- Train a regression model (Support Vector Regression) from jobs with the same name, use the model for predicting the effort.
- Show the jobs to the user, along with their estimated and actual effort and a link to the entire project.
- Allow user to remove projects that do not seem to fit.

Suggested Approach for Similarity Metric Engineering

Suggested Approach



- Foster the strengths of machines and humans: Let
  - machine learn weights
  - humans identify variables

- Let machines and humans help each other in this

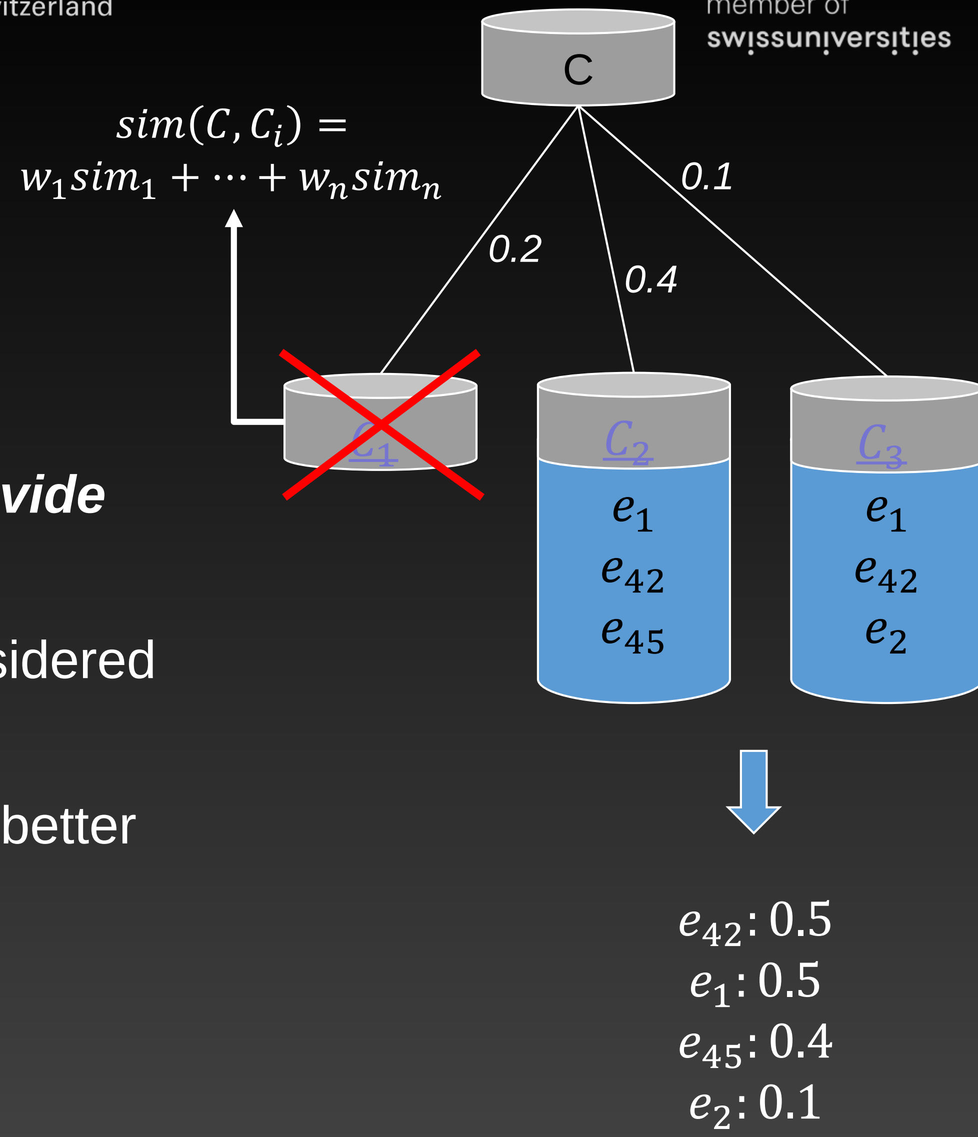
→ This requires to make similarity visible…

Proposed Approach (1)

Example: CBR recommender

Phase 1: adapt weights

- Show similar cases to the user, ***provide access to a case description***
- Allow user to remove ones not considered similar (judged via the description)
- Adapt weights w_i of sim function to better reflect human preference (e.g. via evolutionary algorithm)
- Use selected cases to derive recommendations

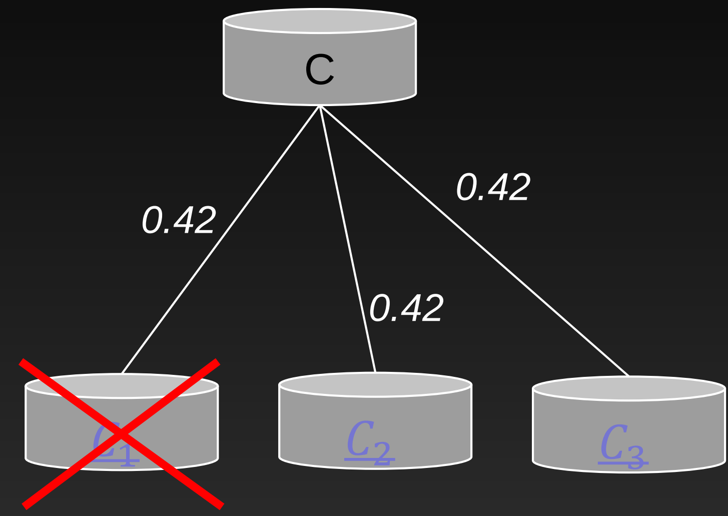


Proposed Approach (2)

Phase 2: include new attributes

1. From logs of phase 1: identify situations where
   1. Retrieved top cases were nearly identical in terms of current attributes, but
   2. Human excluded e.g. one of them

2. Let human experts identify additional attributes that would allow to capture the dissimilarity



Purpose: new webshop area
Client: Doe Inc.
#features: 15
Contact: Jane

Purpose: new webshop area
Client: Doe Inc.
#features: 15
Contact: Joe

Purpose: new webshop area
Client: Doe Inc.
#features: 14
Contact: Joe

Claims for Discussion

Conclusion of Position Paper

Summary: Claims for Discussion

We claim that…

- The approach involves humans and machines in a way that fosters their respective strengths.
- Humans will be motivated to participate because they can better control / improve the results.
- The procedure will increase trust of humans in the final results.

Learning and Engineering Similarity Functions for Business Recommenders

Hans Friedrich Witschel and Andreas Martin