Towards a similarity metric for comparing machine-readable privacy policies

Inger Anne Tøndel and Åsmund Ahlmann Nyre

SINTEF ICT, Trondheim, Norway
{inger.a.tondel, asmund.a.nyre}@sintef.no

Abstract. Current approaches to privacy policy comparison use strict evaluation criteria (e.g. user preferences) and are unable to state how close a given policy is to fulfil these criteria. More flexible approaches for policy comparison is a prerequisite for a number of more advanced privacy services, e.g. improved privacy-enhanced search engines and automatic learning of privacy preferences. This paper describes the challenges related to policy comparison, and outlines what solutions are needed in order to meet these challenges in the context of preference learning privacy agents.

1 Introduction

Internet users commonly encounter situations where they have to decide whether or not to share personal information with service providers. Ideally, users should make such decisions based on the content of the providers' privacy policy. In practice, however, these policies are difficult to read and understand, and are rarely used at all by users [1]. Several technological solutions have been developed to provide privacy advice to users [2–5]. A common approach is to have users specify their privacy preferences and compare these to privacy policies of sites they visit. As an example, the privacy agent AT&T Privacy Bird [2] displays icons to the user based on such a comparison, indicating whether the preferences are met or not. In general, these types of solutions provides a Yes/No answer to whether or not to accept a privacy policy. There is no information on how much the policy differs from the preferences. A policy that is able to fulfil all preferences except for a small deviation on one of the criterion, will result in the same recommendation to the user as a policy that fails to meet all the user's requirements. The user is in most cases informed about the reason for the mismatch, and can judge for himself whether the mismatch is important or not. Still, there are situations where such user involvement is inefficient or impossible, and the similarity assessment must be made automatically.

Automatic comparison of privacy policies is important to be able to give situational *privacy recommendations* to users on the web. The Privacy Finder [3] search engine ranks search results based on their associated privacy practices. Policies are classified according to a predefined set of requirements and grouped into four categories. Thus, sites that are not able to fulfil one of the basic criteria, but offer high privacy protection on other areas will be given a low score. In order to provide more granularity and fair comparisons, a more flexible and accurate similarity metric is needed. Another application area of a similarity metric is for *preference learning in user agents* [6]. This is the application area that we focus on in this paper. To avoid having users manually specify their preferences, machine learning techniques can be utilised to *deduce* users' preferences based on previous decisions and experiences [7]. Thus, having accepted a similar policy before may suggest that the user is inclined to accept this one as well. Evidently, this approach requires a more precise mechanism to determine what constitutes a *similar* policy.

Automatic comparison of privacy policies is particularly complicated due to the subjective nature of privacy [8]. What parts of a policy are most important is dependent on the user attitude and context, and will influence how the similarity metric is to be calculated. In this paper we investigate the difficulties of defining a similarity metric for privacy policy comparison in the context of automatic preference learning. Several privacy policy languages are available, examples being P3P [9], PPL [10] or XACML [11]. Throughout this paper we use P3P in the examples to illustrate challenges as well as potential solutions, but our work is not restricted to P3P. The focus lies on the high-level concepts that need to be solved rather than the particular language dependent problems. The remainder of this paper is organised as follows. Section 2 gives an introduction to Case-Based Reasoning (CBR) and how it can be used to enable user agents to learn users' privacy preferences. Section 3 provides an overview of existing similarity or distance metrics that can be used for comparing policies. Section 4 describes the challenges of policy comparison in more detail, and takes some steps towards a solution. Then Section 5 discusses the implications of our suggestions, before Section 6 concludes the paper.

2 Case-Based Reasoning for privacy

Anna visits a website she has not visited before. Anna's privacy agent tries to retrieve various information on the website, including its machine-readable privacy policy. Then the agent compares its knowledge of the website with its knowledge of Anna's previous user behaviour. In this case, the agent warns Anna that the privacy policy of the website allows wider sharing than what Anna has been known to accept in the past. Anna explains to the agent that she will accept the policy since the service offered is very important to her. The agent subsequently records the decision and explanation to be used for future reference.

Case Based Reasoning (CBR) [12, 13] resembles a form of human reasoning where previously experienced situations (cases) are used to solve new ones. The key idea is to find a stored case that closely resembles the problem at hand, and then adapt the solution of that problem. Figure 1 gives an overview of the main CBR cycle. First, the reasoner retrieves cases that are relevant for the new situation. Then the reasoner selects one or a few cases (a ballpark solution) to use as a starting point for solving the new situation. Then this ballpark solution

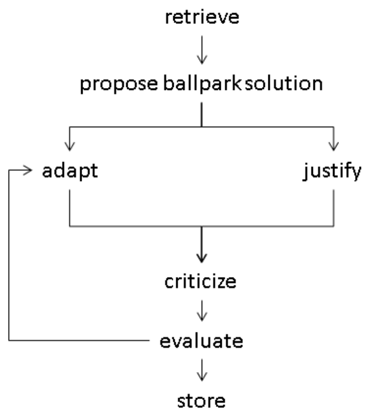

Fig. 1. CBR cycle [12]

is either adapted so that it fits the new situation better, or is used as evidence for or against some solution. The solution or conclusion reached is then criticised before it is evaluated (i.e. tried out) in the real world. It is the feedback that can be gained in the evaluation step that allows the reasoner to learn. In the end, the new case is stored to be used as a basis for future decisions. Central to the CBR approach is the retrieval of *relevant* cases to use as a basis for making decisions. The prevailing retrieval algorithm is *K-Nearest Neighbour* (KNN) [14], which requires a definition of what is consider the *nearest* case. In a privacy policy setting this translates to finding the *most similar* privacy policies, which is the main focus of this paper.

3 Existing distance metrics

There are several existing metrics for computing the difference (or distance) between text strings, vectors, objects and sets and these are often referred to as *distance metrics*. We have so far talked about *similarity metrics* which are really just the inverse of the distance metrics. That is, as the distance increases, the corresponding similarity decreases. However, in order to refer to the different metrics in their original form, we will use the term *distance metric* throughout this section.

Before we survey existing metrics, it is important to clarify what a distance metric actually is. A distance metric is a function d on a set M such that $d : M \times M \to \mathbb{R}$. Where $\mathbb{R}$ is the set of real numbers. Further, the function d must satisfy the following criteria for all $x, y \in M$ [15]:

$$d(x,y) = d(y,x) \qquad \text{(symmetry)} \qquad (1)$$

$$d(x,y) > 0 \iff x \neq y \qquad \text{(non-negative)} \qquad (2)$$

$$d(x,x,) = 0 \qquad \text{(identity)} \qquad (3)$$

$$d(x,y) \leq d(x,z) + d(z,y) \qquad \text{(triangle inequality)} \qquad (4)$$

The symmetry requirement seems obvious, as we normally do not consider the ordering of the objects to compare (whether x or y comes first). As a consequence of this, the second requirement states that there can be no negative distances, since this would indicate a direction and hence the order of comparison would matter. Further it seems obvious that the distance between an object and itself must be zero. The final requirement simply says that the distance between any two objects must always be less than or equal to the distance between the same objects if a detour (via object z) is added. This requirement corresponds to the statement that *"the shortest path between two points is the straight line"*. In the following we introduce three main types of distance metrics; for comparing sets, for comparing vectors or strings, and for comparing objects that are defined through an ontology.

For comparing *sets of objects*, the *Jaccard distance* ($\mathcal{J}_d$) is one alternative metric. It defines the distance between two sets s_1 and s_2 as:

$$\mathcal{J}_d(s_1, s_2) = 1 - \frac{|s_1 \cap s_2|}{|s_1 \cup s_2|} \qquad (5)$$

The metric counts the number of occurrences in the set intersection and divides it by the number of occurrences in the set union. Then it is normalised to return a number in the range $[0,1]$. All occurrences are treated equally, hence the metric does not cater for situations in which some set members are more important than others.

Distance metrics for comparing *vectors or strings* are commonly used to construct error detecting and correcting codes [16]. The predominant such metric is the *Hamming distance*[1], which is defined as *the number of positions in which a source and target vector disagree.* When used in binary representations the Hamming distance can be computed as the

$$d_H(s,t) = wt(s \oplus t) \qquad (6)$$

where the function $wt(v)$ is defined as the number of times the digit 1 occurs in the vector v [16]. However when used in for instance string comparison, the textual definition above must be used. In order to compare vectors or strings of unequal size, the *Levenshtein distance* introduces insertion and deletion as operations to be counted in addition Hamming distance discrepancy count .

[1] Note that the Hamming distance has also been defined for sets ($\mathcal{H}_d$) [17] can be considered a variation of the Jaccard distance, the main difference being that the Hamming distance is not normalised

More formally, it is defined as *the number of operations required to transform a source vector s to a target vector t, where the allowed operations are insertion, deletion and substitution.* Consequently, for equal size vectors, the Levenshtein distance and Hamming distance are identical.

Ontology distances utilise the inherent relationships among objects either explicitly or implicitly defined through an ontology [18]. The approach is used to compute the semantic similarity of objects rather than their textual representation. For example, the distance between *Apple* and *Orange* is shorter than the distance between *Apple* and *House*. In order to determine the distance from one object to another, we can simply count the number of connections in the defining ontology from a source object to a target object via their most recent common ancestor[18].

4 Towards a similarity metric for privacy policies

In this section we start by explaining the various ways in which similarity may be interpreted related to the different parts of privacy policies, and also give an overview of the main parts of the solution needed. Then we make suggestions and present alternatives for creating similarity metrics for comparing individual statements of policies, and also for aggregation of the similarity of statements. Finally we explain how the similarity metrics can be used together with expert knowledge and user interaction in the context of a preference learning user agent.

4.1 What makes privacy policies similar?

In order to be able to automatically determine whether two privacy policies are similar, it is necessary to answer a few basic questions:

- *What makes policies similar?* Can policies offer roughly the same level of protection without having identical practices? And if so, how to determine the level of protection offered?
- *What type of policy content is more important for privacy decisions?* Which policy changes are likely to influence users' privacy decisions? And is it possible to draw conclusions in this respect without consulting the user?

There are surveys available that are able to give some insights into what aspects of privacy are more important to users. As an example, studies performed by Anton et al. in 2002 and 2008 [19] show that Internet users are most concerned about privacy issues related to information transfer, notice/awareness and information storage. However, in order to address the above questions in a satisfactory way in this context, more detailed knowledge is needed.

In this section we start our discussion of similarity by investigating possible interpretations of similarity in the context of P3P policies. Then we outline main issues that need to be addressed in order to arrive at a solution.

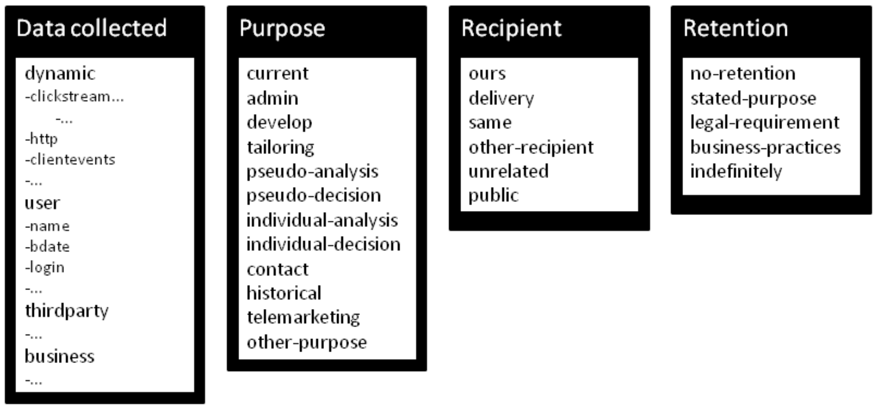

Fig. 2. Alternatives when describing data, purpose, recptients and retention in P3P

Similarity in the context of P3P P3P policies provide, among other things, information on the data handling practices; including the data collected, the purpose of the data collection, the potential recipients of the data and the retention practices. Figure 2 shows the alternatives for describing purpose, recipients and retention using P3P, and also gives an overview of the data types defined in the P3P Base Data Schema [9].

Consider the case where a user wants to compare whether two policies describe collection of similar types of data. Similar could in this context mean that they collect the same data, that they collect a similar number of data items, that they describe data practices that are at the same level in the hierarchy (as an example, `clickstream` and `bdate` is at the same level) or that they collect information that is semantically similar (e.g. part of the same subtree). It could also be possible to add sensitivity levels to data and consider policies to be similar if they collect information of similar sensitivity. For purpose, similarity can mean identical purposes, the same number of purposes or purposes with similar privacy implications. Which similarity interpretation to use it not obvious.

Comparing data handling practices based on their privacy implications is particularly complicated as there is in general no common understanding of the implications of the various practices. Thus, the similarity of for instance the purposes `individual-decision` and `contact` is a matter of opinion - one user may consider `individual-decicion` to be far worse than `contact`, while a second may argue that `contact` is far worse, and a third may consider them to be similar. For recipient and retention it is a bit different, as the alternatives in general can be ordered based on their privacy implications; e.g. `no-retention` is always better than `stated-purpose`, which is again better than `business-practices`, etc. Thus, `stated-purpose` is considered closer to `no-retention` than to `business-practices` The categories are however broad, and e.g. who is included in the recipient group `ours` or `unrelated` will probably vary between policies. The practices of two poli-

Listing 1.1. Excerpt from a P3P privacy policy

```
<STATEMENT>
  <PURPOSE><admin/><develop/></PURPOSE>
  <RECIPIENT><ours/></RECIPIENT>
  <RETENTION><stated-purpose/></RETENTION>
  <DATA-GROUP>
    <DATA ref="#dynamic.clickstream"/>
    <DATA ref="#dynamic.http"/>
  </DATA-GROUP>
</STATEMENT>
```

cies that both share data with `other-recipient` may thus not be considered similar by users.

Comparing full policies further complicates matters. Some users may for instance be most concerned with the amount of information collected, while others are more concerned about the retention practices. When considering the similarity of two policies, it is thus necessary to take into account this variation of importance.

What is needed In order to compare privacy policies and use them in a CBR system we need *similarity metrics* for individual parts of the policy, as well as for entire policies. Such metrics need to be able to handle missing statements, and should also support *similarity weights* to be able to express the criticality or importance of individual statements. Central to the success of the metrics is the ability to understand what similarity means in a given context. Expert knowledge can provide necessary input to the similarity calculations, but as the end-users are experts on their own privacy preferences, it is also important to allow them to influence the similarity calculations.

In CBR, the similarity metric and weight function is normally what is required to compute the k-Nearest Neighbours (i.e. the k most similar policies). Thus, even if there are no policies that would be denoted similar, the algorithm will always return k policies. To cater for this, we require the notion of a similarity threshold such that the algorithm will return only policies that are within the threshold value and that are thus considered to be *similar enough* so that one can be used as a basis to give advice on whether to accept the other.

When using policies to provide advice to users on what to accept or not, it is important to have some understanding of not only the similarity of policies, but also which policy is better or worse. Thus, in addition to a similarity metric, we need a direction vector that can provide this information. This is in part discussed in Section 5, but is considered outside the scope of this paper.

Listing 1.2. Case descriptions

```
datatype=dynamic.clickstream              datatype = dynamic.http,
recipients=['ours']                       recipients = ['ours'],
purpose=['admin', 'develop']              purpose = ['admin', 'develop'],
retention = ['stated-purpose']            retention = ['stated-purpose']
```

Table 1. Overview of local similarity metrics

Attribute	Similarity interpretation	Metric	Input
Data type	Semantic similarity	Ontology	Data schema (+ costs)
Purpose	Equality	Set	(Costs)
Recipients	Privacy implications	Vector/string	(Costs)
Retention	Privacy implications	Vector/string	(Costs)

4.2 Divide and conquer

In our work, we make the assumption that the end-users' preferences when it comes to handling of their personal data is highly dependent on the type of data in question. Thus, we suggest comparing policies with a basis in what data is collected. To illustrate how this can be done we again look to P3P. P3P policies contain one or more statements that explain the handling of particular types of data. Listing 1.1 provides an example of such a statement considering the handling of clickstream data and http data. Due to our data centred approach, we translate such P3P statements into cases (case description) by adding one case per `DATA` item in a statement. As an example, Listing 1.1 contains two `DATA` elements and therefore results in two case descriptions; as given in Listing 1.2.

4.3 Local similarity: Attributes

In common CBR terminology, the term *local* similarity is used when considering the similarity of individual attributes. As already pointed out, similarity may be interpreted in different ways, and in the following we suggest how similarity can be calculated for the attributes data type, purpose, recipients and retention. Table 1 gives an overview of our suggestions. The suggestions have been made taking into account the possible interpretations of similarity and issues related to attribute representation.

The *data type* field is, at least in P3P, based on a data schema that defines a relatively clear semantic relationship between the possible values. Hence it is natural to use an ontology representation and the corresponding *ontology metric* to compute the similarities between these objects. This implies an understanding of similarity as the closeness of the concept. The metric will take into account the distance between the objects as defined in the ontology, resulting in e.g. `family name` being more similar to `given name` than, say, `birthyear`.

For purposes, retention and recipients there is no such data schema available that describes how close the alternatives are to each other. If such a schema is made, this can of course be used in similarity calculations. For *retention and recipient*, however, the alternatives can be said to be ordered, describing practices on a scale from low to high level of privacy-invasiveness. This ordering can be preserved if representing the values as vectors. To illustrate, if we use the ordering, from top to bottom, given in Figure 2 as our basis, we can represent the recipient attribute as a five-dimensional binary vector $v = (v_0, v_1, v_2, v_3, v_4)$ where $v_i = 1$ indicates that the i-th recipient type is present in the attribute. Following this reasoning, the vector representation of the retention attributes given in Listing 1.2 would be $v = (0, 1, 0, 0, 0)$ corresponding to the set representation [`‘stated-purpose’`]. The recipient attribute may have a similar representation, however then using a six-dimensional binary vector corresponding to the six possible values it may take.

For *purposes* it is difficult, not to say impossible, to find an implicit ordering of the possible values. It is for instance difficult to say whether `admin` purpose is far away from `development` purpose, other than the fact that they are different. As a consequence, we believe that the set representation and the corresponding *Jaccard* distance metric are suitable. This implies looking at what purposes are identical.

All of the distance metrics introduced in Section 3 treat all instances equally, and do not take into account that a set or vector member may be more important than another, or that some of the connections in an ontology may be more costly than others. For all the attributes, it is possible to extend the original distance metrics to take into account the cost associated with a difference. This way, a high cost purpose will be more different from a low cost purpose than another high cost purpose, and a jump from e.g. `public` recipient to `unrelated` be rated different than a jump from `delivery` to `ours`.

4.4 Global similarity: Cases

The term *global* similarity is used when aggregating the local similarity values from attribute comparisons to say something about the similarity the entire case descriptions. As our case descriptions are on the level of privacy statements, the global similarity will state the degree to which two statements are similar. Usually, the global similarity is computed by a function that combines the local similarity values.

The most basic of such functions are the average or sum of attribute similarity values. However, since such metrics give equal importance to all attributes, it is more common to use some sort of weighted sum, or weighted average. That way, some attributes may be given more importance (greater relative weight) than others, and therefore will contribute a greater part to the overall similarity assessment. Further, the weight or relative importance of the attributes may be specified by the user or updated on the basis of user feedback, so that these values are also learned by the CBR system.

This may then further be combined with threshold values, such that the global similarity will only be computed if the local similarities are within a predefined threshold. This is to ensure that policies are similar enough to make

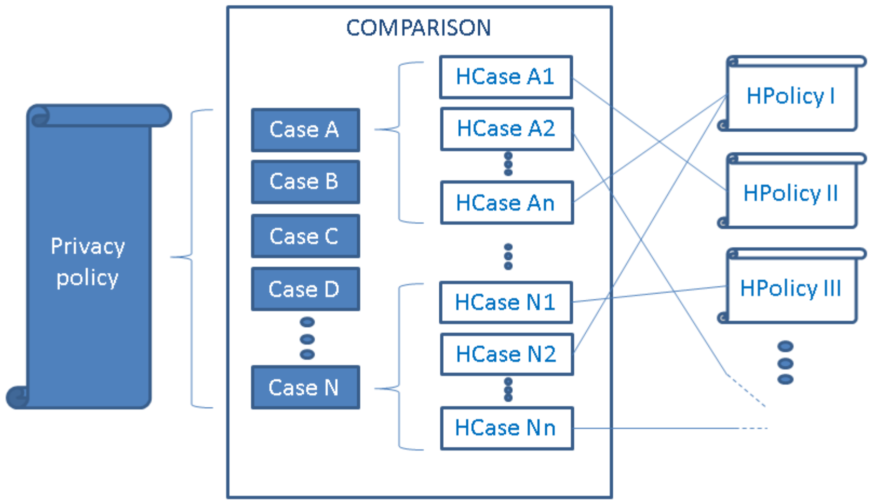

Fig. 3. The cases that belong to one policy may be found to be similar to a number of different historical cases belonging to different historical policies

comparison meaningful and also provide value to the subsequent recommendation made by the CBR system.

4.5 Similarity of policies

Calculating the similarity of cases is not the same as considering the *similarity of policies*, but like for cases, similarity of policies can be computed based on a weighted sum where the weightings taking into account the importance of various data items. But for preference learning user agents, we are not really that interested in comparing full policies. The reason for this is illustrated in Figure 3. As can be seen in this figure, for each individual cases of a policy there is a search for similar historical cases, and the cases that are a result of these searches may belong to a number of different policies, originally. Comparing all these historical policies to the current privacy policy is not necessarily useful. Instead it is important to determine, based on these similar cases, what advice to provide to the user. Thus it is more important to consider whether or not the user has accepted this kind of practice (as described by the case) in the past. For each case of a policy it should be possible to reach a conclusion about whether or not the user is likely or not to accept this practice (e.g. either yes, no or indeterminate). Then, to reach a conclusion about the policy, the total likelihood of acceptance could be computed based on a weighted sum taking into account the importance of each case (based on the type of data it concerns).

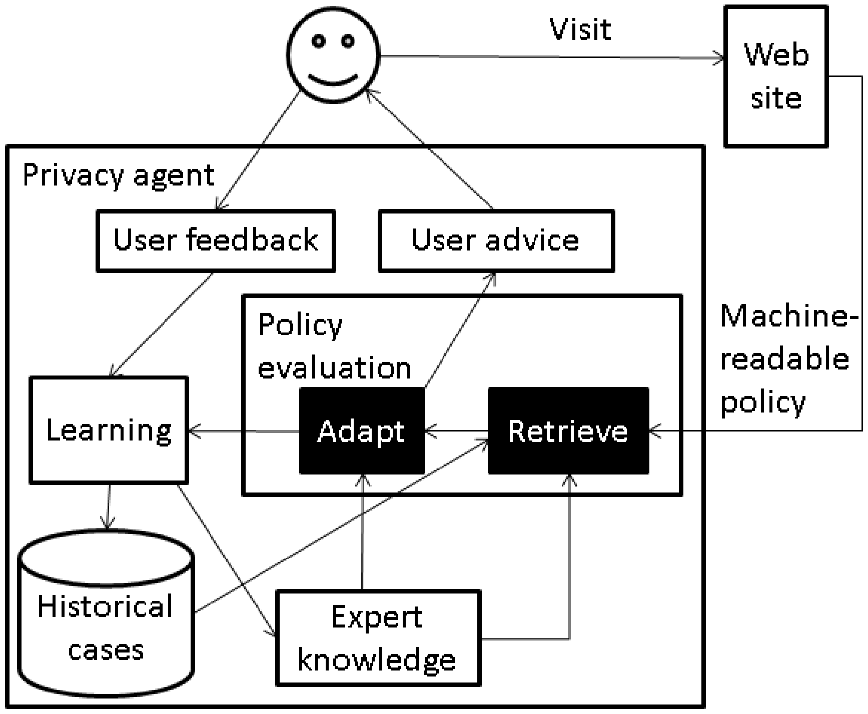

Fig. 4. Applying similarity metrics in preference learning user agents

4.6 Applying similarity metrics in preference learning user agent

Figure 4 gives an overview of the type of solution we envision for policy comparison in the context of a privacy agent. When the user visits a website, the machine-readable policy of this website is used as a basis for providing the user with recommendations as to whether or not to share personal information with this site. During policy evaluation the new policy is divided into a number of cases, based on the data collected, and each of these cases are evaluated towards the historical cases. The historical cases that are most similar to the current situation are used to come to a conclusion on what recommendation to give the user.

In order to be able to retrieve similar cases, the similarity metric is used together with the similarity weight function and the similarity threshold. The necessary input to the similarity calculations, such as costs and weights, are included as expert knowledge. Expert knowledge also provide information on what alternatives are better or worse in terms of privacy protection. Privacy experts will be in the best position to provide this type of information, and let users benefit from their expertise. However, what is considered to be the most important privacy concepts will likely vary between user groups, and also individuals. It can also be dependent on the legal jurisdiction [20]. Expert knowledge can be specified in a way that takes into account some of these likely variations. However, it is also possible to make solutions that allow users to influence the expert knowledge that is used as a basis for making recommendations. We will come back to this shortly.

When the most similar cases have been retrieved, these are combined and adapted in order to come to a conclusion regarding the current situations. As already explained, this can be done by by simply taking a weighted majority-vote based on the set of cases selected. However, as the agent should be able to explain its reasoning to the user, there is also a need to build an argument that can be used to explain the agent's decision. In this process, expert knowledge also has a role to play by e.g. explaining why something is important.

The conclusion reached is presented to the user through for example a warning to the user of problems with the policy, or no warning if the policy is likely to be accepted by the user. The user, in the same way, may or may not provide feedback on this decision, e.g. by stating that he disagrees with the reasoning behind the warning. Either way, the acceptance or correction of the recommendation given is important and makes the agent able to learn and thereby improve its reasoning. A correction of the agent's reasoning may trigger a re-evaluation of the policy, and result in updates to the current cases that are stored in the case repository. But the correction can also, at least in some cases, be used to improve the expert knowledge used in the policy evaluation. After all, users are experts on their own privacy preferences, and can make corrections of type "I do not care who gets my email address", or "I will never allow telemarketing, no matter the benefits"

5 Discussion

In this paper we have shown how privacy policies can be divided into a number of cases that can then be compared to other cases individually. We have also proposed what type of similarity metric to use to compare attributes of cases, and shown how the results of these individual comparisons can be used to say something about the similarity of cases and policies at a higher level. In this section we discuss important parts of our suggestions, focusing mainly on areas where further research is needed.

5.1 The role of the expert

Existing distance metrics can be modified to take into account costs, but for this to work we need a way to determine these varying costs. As has already been pointed out previously in this paper, deciding what is better or worse when it comes to privacy, and how much better or worse it is, is very much a matter of opinion rather than facts. Privacy experts are the ones most capable of making such statements when it comes to cost, but further research may be needed in order to agree on useful cost values. The same goes for weights that are used for computing global similarity values, and for making recommendations to users.

5.2 Involving the end-user

In our suggestions for applying similarity metrics for preference learning user agents, we have emphasised the need to include user input and use this input to improve the similarity measures. For this to work, there is a need for good user interfaces and also a need to understand what users will be able to understand and communicate related to their preferences. A key dilemma is to find the right level of user involvement. It is important to involve users in the learning process, but if users receive a lot of requests for feedback on similarity calculations, this may be considered to be annoying interruptions and will likely result in users refraining from using the agent. It is also important to find ways to weigh the opinions of the users against the expert knowledge.

5.3 Differences between policy languages

In this paper we have used P3P as an example language, and the metrics suggested have been discussed based on the way policies are represented in P3P. The metrics selected may be different if policies are presented using other languages. As an example, the PPL specifications [10] show examples where retention is specified using days rather than the type of practice. This will result in the use of a different type of metric, e.g. the Euclidean distance. However, how would you compare a retention of 30 days with, say, `business-practices`? Here, again, experts are the ones that can contribute with knowledge on how to solve this, but to gain such explicit knowledge and agree on the necessary parameters will likely require further research.

5.4 Aggregation of similarity values

Though this paper provides some suggestions as to how the similarity of cases and policies can be computed, more work is needed on this topic. The examples we use only consider parts of the P3P policies. In addition, there will in many cases be a need to take into account the direction of the difference (better/worse). This direction cannot be included directly in the similarity metric, as this would violate the symmetry criteria in the very definition of a distance metric. Still the distance is important when making choices based on the result of a metric.

For preference learning privacy agents, the policy will only be one of several factors to consider when making recommendations to users. Additional factors include context information and community input [6]. This complicates the similarity calculations.

5.5 CBR vs. policy comparison in general

Up till now we have mainly discussed the problems related to policy comparison in a situation where historical decisions on policies are used to determine what recommendations to give to users in new situations. However, in the introduction we pointed at other types of applications where automatic policy comparison can be useful, e.g. in privacy-aware search engines. So, how do our suggestions relate to such other uses?

We have considered situations in which privacy policies are compared with policies the user has accepted or rejected previously, but this is not that different from comparing a policy to those of similar types of sites to e.g. find how a web shop's privacy practices are compared to other web shops. In both cases there is no strict pre-specified matching criteria to use. Expert knowledge will be important in both cases, to assess what aspects of a policy are better or worse, and how much better or worse. But where we have been mainly concerned with identifying similar cases, other uses may be more interested in identifying policies that offer better protection, and say something about how much better this protection is. Case retrieval will be different in such settings, as it is important to consider all cases belonging to one policy together. There is also a need to calculate the similarity of policies, and not only cases.

6 Conclusion and further work

In order to develop new and improved privacy services that can compare privacy policies in a more flexible manner than today, there is a need to develop a similarity metric that can be used to calculate how much better or worse one policy is compared to another. This paper provides some steps towards such a similarity metric for privacy policies. It proposes similarity metrics for individual parts of policies, and also addresses how these local similarity metrics can be used to compute more global similarity values. Expert knowledge serves as important inputs to the metrics.

In our future work we plan on implementing measures for policy comparison in the context of preference learning user agents. The similarity metrics will be evaluated by comparing the similarity values that are automatically computed with the values stated by users, when asked. The input received will also be used to improve the expert knowledge that the calculations rely on.

Acknowledgments

We want to thank our colleague Karin Bernsmed for useful input in the discussions leading up to this paper.

References

1. C. Jensen, C. Potts, and C. Jensen, "Privacy practices of internet users: Self-reports versus observed behavior," *International Journal of Human-Computer Studies*, vol. 63, no. 1-2, pp. 203 – 227, 2005.
2. L. F. Cranor, P. Guduru, and M. Arjula, "User interfaces for privacy agents," *ACM Trans. Comput.-Hum. Interact.*, vol. 13, no. 2, pp. 135–178, 2006.
3. "Privacy Finder. http://www.privacyfinder.org."

4. J. Camenisch, A. Shelat, D. Sommer, S. Fischer-Hübner, M. Hansen, H. Krasemann, G. Lacoste, R. Leenes, and J. Tseng, "Privacy and identity management for everyone," in *Proceedings of the 2005 workshop on Digital identity management*, ser. DIM '05, 2005, pp. 20–27.
5. S. E. Levy and C. Gutwin, "Improving understanding of website privacy policies with fine-grained policy anchors," in *Proceedings of the 14th international conference on World Wide Web*, ser. WWW '05, 2005, pp. 480–488.
6. I. A. Tøndel, Å. A. Nyre, and K. Bernsmed, "Learning privacy preferences," in *Proceedings of the Sixth International Conference on Availability, Reliability and Security (ARES 2011)*, 2011.
7. B. Berendt, O. Günther, and S. Spiekermann, "Privacy in e-commerce: stated preferences vs. actual behavior," *Commun. ACM*, vol. 48, no. 4, pp. 101–106, 2005.
8. S. A. Bagüés, L. A. R. Surutusa, M. Arias, C. Fernández-Valdivelso, and I. R. Matías, "Personal privacy management for common users," *International Journal of Smart Home*, vol. 3, no. 2, pp. 89–106, 2009.
9. "W3C. Platform for Privacy Preferences. http://www.w3.org/P3P/."
10. S. Trabelsi, "Second release of the policy engine," Prime Life, Tech. Rep. D5.3.2, 2010.
11. "OASIS eXtensible Access Control Markup Language (XACML). http://www.oasis-open.org/committees/xacml."
12. J. L. Kolodner, "An introduction to case-based reasoning," *Artificial Intelligence Review*, vol. 6, pp. 3–34, 1992.
13. A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issues, methodological variations, and system approaches," *AI Communications*, vol. 7, no. 1, pp. 39–59, March 1994.
14. T. Mitchell, *Machine Learning*. McGraw Hill, 1997.
15. T. Bozkaya and M. Ozsoyoglu, "Distance-based indexing for high-dimensional metric spaces," *SIGMOD Rec.*, vol. 26, pp. 357–368, June 1997.
16. D. C. Hankerson, G. Hoffman, D. A. Leonard, C. C. Lindner, K. T. Phelps, C. A. Rodger, and J. R. Wall, *Coding Theory and Cryptography: The Essentials*, 2nd ed. New York, NY, USA: Marcel Dekker, Inc., 2000.
17. A. Arasu, V. Ganti, and R. Kaushik, "Efficient exact set-similarity joins," in *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB)*. VLDB Endowment, September 2006, pp. 918–929.
18. A. Bernstein, E. Kaufmann, C. Bürki, and M. Klein, "How similar is it? towards personalized similarity measures in ontologies," in *Wirtschaftsinformatik 2005*, O. K. Ferstl, E. J. Sinz, S. Eckert, and T. Isselhorst, Eds. Physica-Verlag HD, 2005, pp. 1347–1366.
19. A. I. Anton, J. B. Earp, and J. D. Young, "How internet users' privacy concerns have evolved since 2002," *IEEE Security and Privacy*, vol. 8, pp. 21–27, 2010.
20. S. Fischer-Hübner, E. Wästlund, and H. Zwingelberg, "Ui prototypes: Policy administration and presentation version 1," Prime Life, Tech. Rep. D4.3.1, 2009.