Cross training and its application to skill mining

by D. A. Oblinger
M. Reid
M. Brodie
R. de Salvo Braz

We present an approach for cataloging an organization's skill assets based on electronic communications. Our approach trains classifiers using messages from skill-related discussion groups and then applies those classifiers to a different distribution of person-related e-mail messages. We present a general framework, called cross training, for addressing such discrepancies between the training and test distributions. We outline two instances of the general cross-training problem, develop algorithms for each, and empirically demonstrate the efficacy of our solution in the skill-mining context.

An increasing number of knowledge-intensive organizations prize their human skill set as their primary asset. Like any asset, skills can only be fully utilized if they are well understood and carefully cataloged. The ability to locate people with specified skills is crucial for many business activities, including managing projects, answering questions, and building project teams. An organization-level catalog of employee skills serves as the basis for identifying skill gaps, and in quantifying an organization's value.[1–3]

Skill assets present two difficulties beyond those associated with conventional assets. First, because each individual has a large and unique set of skills, the size of the skill catalogs for an organization is quite large. Second, characterization of skill assets is often more subjective than conventional assets. Both of these difficulties make the creation and continuous maintenance of this catalog over time an exceedingly expensive operation. This expense and the importance of the information itself spurred our investigation of automated methods for addressing the problem.

Any automatic means of creating and maintaining a skills catalog must rely on a data source with several specific properties. (1) It must be inexpensive to access electronically—a cost-effective solution cannot rely on expensive data. (2) The data must be ubiquitous and inclusive—our goal is a comprehensive skill catalog, thus our data source must comprehensively cover an organization's employees. (3) The data source must contain rich personal data—a person's skills are multifaceted, and capturing a meaningful piece of this picture will require rich data about that person. (4) The data source must be current and continuously updated, because we are interested in maintaining a skill catalog over time.

For many organizations the only data source that satisfies all four of our requirements is personal electronic communications. This includes e-mail, postings into public or private discussion groups, and documents authored by the individual. In this paper we refer collectively to these forms of person-related content as *personal communications* or person-related messages. We refer to the problem addressed in this paper as the *skill-mining problem*, that is, the problem of creating and maintaining a skill catalog,

a mapping from a set of employees to a set of business-relevant skill categories.

We tackle the skill-mining problem using inductive learning techniques to induce a skill detector for each business-relevant skill and then apply these to an organization's personal communications. The skill detector must make a single person-level prediction using the set of messages related to that person. However, it is difficult to obtain training data for which a single label is assigned to a collection of messages. The labeled training data that are naturally available for this task are individual skill-related messages.

Thus successful skill mining requires dealing with a mismatch between the training and test distributions. This mismatch stretches the traditional inductive learning framework, which assumes that both the training and testing data are obtained from a single distribution.[4] We define a generalization of the conventional inductive learning framework, called cross training, to address this mismatch between the training and test distributions. We propose solutions for specific instances of the cross-training framework and show how these can be applied to the skill-mining problem.

In the next section we formulate the skill-mining problem precisely and explain the nature of the mismatch between the training and test distributions. The following section introduces the cross-training framework and proposes algorithms for dealing with specific instances of this framework. Following sections describe our implementation of these methods to address the skill-mining problem, present experimental results that show that taking advantage of the nature of the mismatch gives good learning performance, and provide an analysis of our results. In the final sections we discuss related work and contemplate future work.

Skill-mining architecture

In many ways our approach to skill mining is like many conventional text-mining approaches to message classification.[5] Thus in the next subsection we highlight only those aspects that differ.

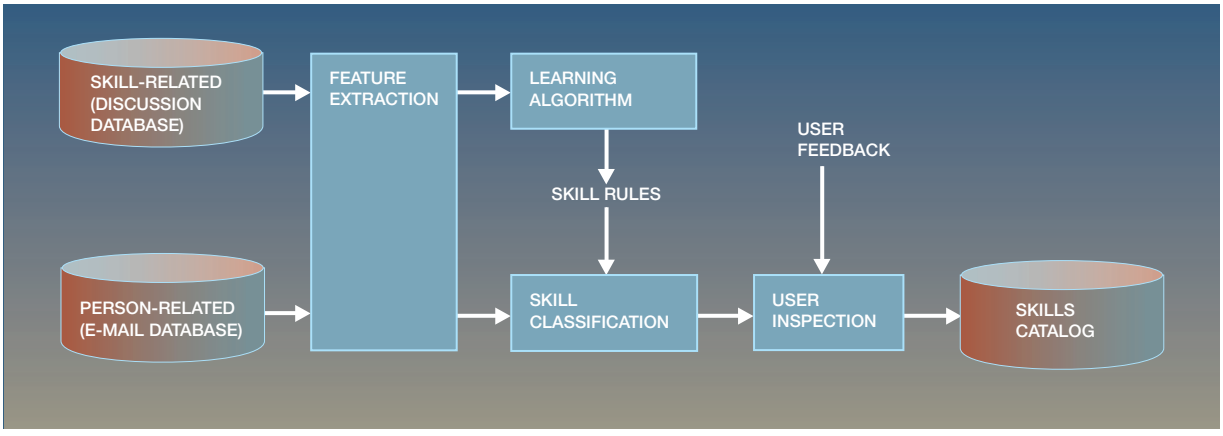
Data sources. Training our skill recognizers requires data labeled by business-relevant skill categories. We employ Usenet and internal corporate discussion databases, organized by skill topic, as our primary sources. These data sources not only contain messages similar to our target personal communication data but also have the advantage of being relatively "on-topic" within each discussion group. It is important to note that each *message* from these sources has a skill label, whereas in the target prediction task each *person* is assigned a skill label. Because we are representing persons by the set of messages they author or receive, our skill recognizers are trained on individual messages but must classify sets of messages. This discrepancy is significant, and our solution for it is detailed later.

Feature extraction. Many inductive learning techniques require training and testing instances to be represented as a fixed vector of features. We adopt the common approach of reducing short, free-text messages to a "bag of words."[5–7] In this representation, each position in the vector has a binary value that represents the occurrence or nonoccurrence of a specified word in the training or testing message. We limit ourselves to binary features because of the discrepancies between training and target distributions. We have observed that public discussion messages and point-to-point messages (like e-mail) differ in a number of ways; for instance, e-mail messages are typically shorter and have a greater implied context, using pronouns in place of proper nouns. Both of these effects skew the expected feature counts. Compression into binary features lessens the sensitivity to such discrepancies.

Feature context. An aspect of the skill-mining problem that differentiates it from many text-mining problems is the rich word context available. The occurrence of each word in an electronic communication has two types of context. The first context is the way a word is positioned within a message. "Present in the subject field," "used in a quote when replying," or simply "part of the body" are all possible word contexts. A second type of feature context relates to the function of the message the word appears in. "Sent to a distribution list," "sent as a reply," or "received message" are examples of this second type of context.

Each word can therefore give rise to several binary features by taking into account its context. For example, the programming-related term "hash table" would be expressed as a series of features such as "hash table present in the subject field" and "hash-table in the body of a message sent to a distribution list." Making these distinctions can be quite important, because the relevance of a term to a person's skill depends on both the vocabulary of a skill area

Figure 1 An overview of the skill-mining process



and how the person uses that vocabulary—was the word used in response to an e-mail message, or merely received in a mass-mailed announcement?

In our work to date, we have fixed the number and type of contexts used, guessing at a balance between the expressivity of our hypothesis space and its complexity. An interesting direction for future work is to consider dynamically introducing various contexts in an iterative way, thus gaining benefits of more expressive classifier rules while not blindly multiplying the features used in representing the problem.

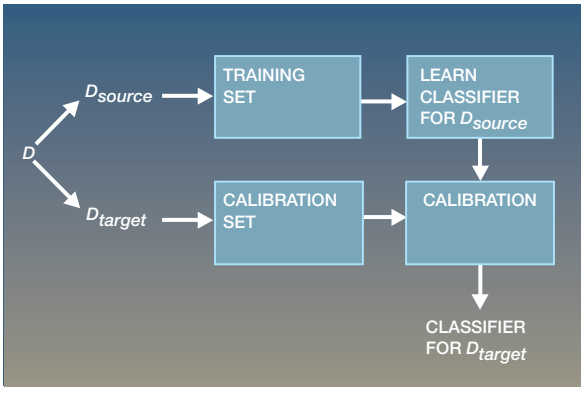
Prediction inspection as a solution for data privacy. A crucial social issue regarding personal communication data, e-mail in particular, is privacy. E-mail is an extremely rich source of up-to-date, personal skill data, but at the same time ethical considerations, and most corporate policies, prohibit publishing those data. Skill mining provides a solution to this problem by compressing private, person-related data into a concise skill profile. This skill summary is concise enough to make inspection by the individual being profiled feasible. Summarization and subsequent inspection is therefore an essential step in making use of personal communications as a data source.

Skill-mining architecture. The skill-mining process is summarized in Figure 1. The skill-related and person-related data are compressed into training and testing instances. Classifiers are induced from the former, transformed into skill rules, and applied to the latter, resulting in a skill profile for each employee. Because employee skills form overlapping sets (each employee will simultaneously have many skills) we express the skill classification problem as a set of binary classification problems, one for each skill category. Once all classifications have been determined for a given employee, he or she can then examine the profile to determine which portions are to be published. Then it is stored for use in subsequent knowledge management tasks.

Induction of person-based skill classifiers. The problem of learning skill classifiers seems, at first, just like a typical problem of learning from unstructured text. We have a corpus of labeled examples and testing data, both of which consist of similar kinds of text messages. So it should just be a matter of applying our favorite learning algorithm to the training data and using the results to classify the test data.

However, if we look a bit closer we see two very important problems with this naive approach. First, most of the messages contained in the testing set (messages to friends, jokes, company-wide memos) do not correspond to any category listed in the training set (technical information found in discussion databases). Second, the skill labels in the training data are assigned on a message-by-message basis; however, the required skill predictions are on a person-by-person basis. Effectively, this means that we must classify an individual's entire message database with a single label denoting that person's predicted skill level.

Figure 2 The cross-training framework



A natural approach for the second problem would be to concatenate all person-related mail for an individual into a single "message" and try to predict labels for it. This approach yields an enormous message, and such a large disparity in message size is not consistent with our representation of messages as a bag of words or our use of binary features. In both cases the large size exacerbates the information loss in our representation. One might consider other representations that are not as sensitive to message size, but we believe that simply adopting an approach less sensitive to this difference will not be as strong as explicitly compensating for the difference. A method for explicitly compensating for this problem is outlined in the next section.

The simplest approach to making the training data more amenable to the test data would be to inject some source of representative noise into the training set. This would be very damaging, because an average person's message database consists almost entirely of skill-irrelevant messages (above 95 percent). This high level of artificially injected noise would overwhelm any learning algorithm. The next section presents our approach for compensating for this difference between training and test data.

Cross training

Before we present a specific solution to the skill-mining problem, we take a step back and try to characterize the problem more generally. This characterization not only allows this and prior work to be expressed in a common framework; it also provides a context for future work.

The cross-training framework. Traditional classification learning assumes that learning is performed on a domain expressed as a probability distribution D over pairs $(e, 1)$ of examples, e, and their labels, 1. Samples of the training and testing sets are drawn independently from D.[4,8] We define *cross training* as a generalization of traditional classification in which the training and testing sets are drawn from two different distributions, denoted by D_{source} and D_{target}. We assume that a fixed but unknown process has generated the source and target distributions from some underlying distribution D.

Figure 2 shows the general cross-training framework. Learning proceeds in two stages: (1) conventional learning is performed over D_{source}, producing a classifier CL_1. (2) A second phase, which we call "calibration," is then performed using a small amount of data drawn from the D_{target} distribution, yielding a classifier CL_2 for the target domain. Of course any testing of CL_2 is done using a completely independent test set drawn from D_{target}.

Our interest in this particular framework will be obvious to the machine-learning practitioner who has encountered learning tasks where either the training or testing data have been systematically perturbed. For example, in practical applications a system that has been trained in one environment (e.g., the help desk at central headquarters) is often installed and used in a related, but different environment (e.g., the help desk at a local branch office). As another illustration, there may be different underlying processes that are responsible for generating the positive and negative examples in the training set (e.g., in medical diagnosis, profiles of patients with and without disease are often obtained from different sources) that are not reflected in the test environment.

There is no general solution to the cross-training problem—any number of relationships might exist between D_{source} and D_{target}. Some problems that have already been addressed in the machine-learning literature can be formulated in the cross-training framework—we discuss this further in a later section. In this section we explore the two instances of the cross-training framework that are relevant to the skill-mining task. We provide algorithms that are designed to take advantage of these relationships. The increased complexity of this methodology is justified by the superior results that this approach yields, as shown experimentally in a following section.

Dilution cross training. As the name suggests, the target domain is here a "watered-down" version of the training domain. This means D_{target} may contain instances for which labels do not appear in D_{source}. To simplify the discussion we assume all such instances come from the same class, called "other." We also assume that dilution does not affect the relative proportions of the classes in D_{target} when compared with D_{source}. This is done to clearly separate the issue of dilution cross training from the issue of unbalanced class distributions, which we discuss further under related work.

We define a cross-training problem's *dilution rate* to be the probability of drawing an instance labeled "other" from D_{target}. An obvious approach to dealing with diluted domains is to simply ignore the "other" class and to apply a traditional learning algorithm. Since many learning algorithms can handle small amounts of noise, this approach will yield good results when the dilution rate is relatively small. Thus we refer to a problem as a dilution cross-training problem only in the case that it has moderate-to-high dilution rates.

To build a classifier for a diluted domain we will require that any classifier for the training domain must return a *score* indicating the relative strength of its predicted class label. Many standard inductive learning algorithms are readily adapted to this extended form: support vector machines, Bayesian algorithms, decision-tree learners, and neural networks all can return a prediction score. In the case of decision-tree learners and Bayesian algorithms, the score provided is, in fact, the estimated probability of class membership. The score returned by support vector machines and neural networks has a less clear interpretation. Nonetheless, all we require of this score is monotonicity, that is, a higher score indicates a higher estimated probability of membership in the predicted class.

We define a threshold t_d that allows a classifier CL_1 for D_{source} to be transformed into a classifier CL_2 for a diluted domain D_{target} as shown in Figure 3.

If we consider the target domain as having only two classes, "other" and "important" (an amalgamation of the classes in the training domain), then the membership threshold will control false positive and false negative rates for the "important" class. Calibrating the classifier is now a question of selecting an appropriate value for the membership threshold. In most cases, the false positive and false negative rates will vary smoothly with t_d. This means a simple optimization technique, such as gradient descent, can be used over the calibration set to determine an optimal value for this parameter.

Figure 3 Dilution cross training

Given: CL_1 a classifier for D_{source}
Define:
$CL_2(e) = I$ if $CL_1(e) = I$ with score $s > t_d$
$CL_2(e) =$ "other" otherwise
where e is any instance from D_{target}

Grain-size cross training. The grain-size issue arises when the instances from D_{target} are actually collections of smaller objects in D_{source}, and learning a classifier for these constituent objects is easier than learning a classifier for the entire collection. We assume that the label of the entire collection is a function of the labels of the constituents. The *grain size* refers to how many smaller objects from D_{source} make up each instance from D_{target}.

As an example of this, suppose we have a classifier trained to recognize the presence of any one of a set of objects in a scene, and we wish to detect which objects are present in a continuous video stream. If the objects (or camera) are moving, we cannot simply combine the images in the stream into a single image. We need to modulate the predictions made on each image in the stream in order to generate a single prediction for the entire stream.

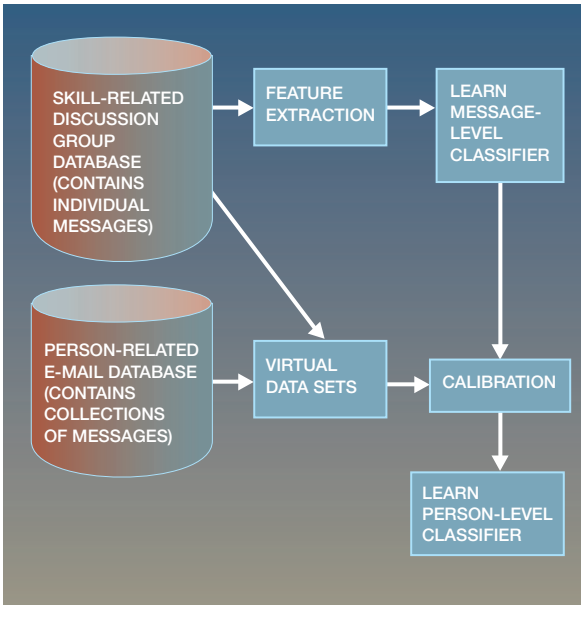
A simple approach to calibration for differing grain sizes is to try to determine how many grains in the collection need to have a high-enough score of predicted class membership in order for the collection to be in that class. More precisely, we define two thresholds t_{g1} and t_{g2} that allow a classifier CL_1 for D_{source} to be transformed into a classifier CL_2 for a domain D_{target} of different grain size, as shown in Figure 4 (for simplicity we assume two classes).

This reduces the problem of cross training a classifier between domains with differing granularity to finding a good choice for the parameter values for t_{g1} and t_{g2} that yield good performance on the calibration set. As an example, suppose we wish to recognize whether or not a particular object is present in a sequence of video frames, and we have trained

Figure 4 Grain size cross training; classifier CL_2 estimates how many elements need to score well according to CL_1.

Given: CL_1 a classifier for D_{source}
$CL_1(o)$ is a score denoting the estimated strength of class membership o.
Let $e = \{e_1,...e_n\}$ be any instance from D_{target}
Define: $CL_2(e) = |\{e_i,: CL(e_i) > t_{g1}\}| > t_{g2}$

Figure 5 Skill-mining implementation of cross training



a classifier that (imperfectly) detects the presence of the object in any particular frame. Optimally setting t_{g1} and t_{g2} for this problem will yield a classifier that relies on the redundancy in the sequence to robustly detect the presence of the object.

Skill-mining implementation

Recall that the skill-mining problem is to use training data consisting of individual labeled messages taken from skill-specific discussion groups to induce a classifier for entire collections of personal e-mails. This presents a combination of dilution and grain-size problems. Dilution occurs because most of the messages in the testing set (e.g., messages to friends, jokes, company-wide memos) do not correspond to any label in the training set (specific skills from discussion groups). The grain-size problem occurs because the skill labels in the training data are assigned to individual messages, whereas the required skill predictions must be made on collections of messages.

When testing our solution to the cross-training problem, we would like to explore its ability to compensate for dilution and grain-size effects. We have skill-related data in the form of over 30000 skill-related discussion group messages from over 60 skill categories, as well as person-related data in the form of entire e-mail databases for 16 individuals, averaging about 10000 messages per e-mail database. These data sets display pronounced dilution and grain-size effects. However, it is not possible to control the dilution or grain-size effects within the available person-related data. Indeed, in the case of dilution it is not even possible to quantify the amount of dilution occurring, since we do not have any labeling at the level of individual e-mail messages.

To address these limitations, we describe a method for constructing virtual data sets with known and controllable dilution rates and grain size. Each virtual data set will be a collection of messages representing an individual, and the final cross-trained classifier will classify these virtual data sets.

Figure 5 shows the skill-mining architecture. First a message-level classifier must be learned on the skill-related messages from the discussion groups. This is done in two steps: extraction to obtain a good set of features, followed by induction of a simple classifier. Then this classifier must be "calibrated" using the virtual data sets to create a person-level classifier. We now describe each module in detail.

Feature extraction. We adopt the common approach of representing each message by a fixed vector, using the bag of words approach.[6,7] In this representation, each position in the vector has a binary value that represents the occurrence or nonoccurrence in

the message of a specific word. We limit ourselves to binary features because of the discrepancies that exist between skill-related discussion messages and personal communication messages. Among the differences we have observed are: (1) personal communication messages are typically shorter; (2) personal communications tend to use pronouns in place of proper nouns more often; (3) words in the messages have different contexts—the location of a word within a message, or the function of the message in which the word appears. These differences skew the expected feature counts. Compression into binary features lessens the sensitivity to such discrepancies.

Learning a message-level classifier. Because of the systematic differences just described, we need a message-level classifier that makes few assumptions regarding the specific structure of the messages themselves, so we focus on the most predictive, isolated features. We order the terms by information gain, because the terms with highest information gain tend to be jargon terms associated with a skill category.

For example, the sidebar shows the best 50 features, ordered by information gain, for the "Java programming" skill category. Any Java** programmer will recognize the majority of words in this list as an essential part of his or her vocabulary (although some are the names of frequent posters to the Java discussion list!). Using mutual information instead of information gain produces a very similar list, whereas using the raw word count results in more terms that are not jargon being ranked highly.

Given that our data set has over 100 000 unique word tokens, using a list such as this to drive some form of feature selection is essential. On the other hand, we did not find that compensating for strong interactions between jargon terms was necessary for identifying skill messages with high reliability, so we employ an "*m*-of-*n*" concept in our message-level classifier. Given this ordered skill-related word list, the classifier can be expressed as a pair of integers. A message is considered to be a skill message if at least m of the first n listed words occurs in that message. This classifier is both sufficient for this domain and allows us to focus on the effects of cross training. The framework itself, of course, admits the use of any classifier over the source distribution.

Virtual data sets. For empirical evaluation we require a source of testing data that has known and controllable dilution and grain-size effects. We obtain this by using both the discussion groups and the e-mail

The 50 terms with the largest information gain for the Java programming category

class
classes
applet
jdk
methods
hicks
applets
void
classpath
paradine
jvm
warwick
swing
stan
jar
jni
awt
reuse
implementations
wilde
args
rmi
vajava
paint
bean
javac
boolean
unicode
javaos2
feinberg
constructor
beans
christensen
vaj
bolmarcich
malte
subclass
throws
kress
jre
appletviewer
hisey
hotjava
inner
mik
extpkg
dialogs
beth
kesselman
locale

Figure 6 The cross-trained classifier for skill mining. Classification at the person level is determined by estimating how many individual messages need to be sufficiently skill-related. The calibration set is used to find optimal values for m, n, and o.

Given: W = list of terms ordered by information gain
Let m, n, o be positive integers.
For any message M, let $|M^n|$ denote the number of words appearing in both M and the first n words W.
Let $e = |M_r|$ be a collection of messages.
Define $|e^n{}_m|$ = the number of messages in e with $|M^n| > m$.
Then e is classified positive if and only if $|e^n{}_m| > o$.

databases to create virtual data sets with known dilution rate and grain size.

To simulate a diluted test set, we first require messages that are not relevant to the skill we are trying to identify. Some of the 16 e-mail databases belong to individuals who, when surveyed, reported that they were not experts in any of the skill categories. We therefore assume that the messages in these individuals' databases are not skill-related. Public discussion forums that are unrelated to the skills in question also serve as a source of skill-irrelevant messages. Collectively, these form a pool of over 30000 skill-irrelevant messages from which we can draw in order to controllably dilute a set of skill-related messages.

As an example, constructing a virtual data set with 1000 messages and a dilution rate of 99 percent is simply a matter of choosing 990 messages from the pool of skill-irrelevant messages and ten messages from the pool of skill-related messages. By varying the percentage of skill-related messages we can control the dilution rate.

The grain size can be controlled by varying the total number of messages that make up a single virtual data set. Thus a set of virtual data sets can be generated with any dilution rate D and grain size G.

Cross training. The cross-trained classifier operates on *collections* of messages. The calibration set is a small sample of virtual data sets. A virtual data set is regarded as "negative" if it is composed entirely of skill-irrelevant messages and "positive" if it contains as many skill-irrelevant messages as indicated by the dilution rate. For example, a dilution rate of 99 percent means that positive data sets have 1 percent skill-related messages and 99 percent skill-irrelevant messages.

Given a collection of messages, cross training estimates the "correct number" of individual messages that need to get a "high-enough" score from the message-level classifier. More precisely, let m, n, and o be positive integers. For each message, we determine if at least m of the n most-informative features occur in that message. A collection of messages is classified as positive if at least o of the messages satisfy this m-of-n threshold. This is summarized in Figure 6. A simple hill-climbing search is performed to find values for the thresholds m, n, and o that give the best performance on the calibration set.

The cross-trained classifier is tested on an independently generated sample of virtual data sets. Performance is measured as the percentage of the data sets in this testing set that the system classifies correctly. Notice that we are not scoring the system on its ability to classify the messages themselves but rather the ability to correctly classify the group of messages associated with a single individual.

Experimental results

Our goal is to measure the performance of cross training as applied to the skill-mining problem. For this experiment we chose the "Java programming" discussion group as our skill category, thus we are plotting the system's ability to recognize persons whose electronic communications involve discussions of Java programming. Several thousand skill-related messages drawn from the Java discussion groups and about 30000 skill-irrelevant messages drawn from about 60 discussion groups not related to Java are used to construct the message-level classifier.

Experiment 1: Uncalibrated learning. We first use the virtual data sets to directly test the naive approach of simply concatenating all of the messages in a virtual data set into a single message and using the message-level classifier. No correction was made to compensate for the larger message sizes or the shift in probabilities. As one might expect, the performance of this uncalibrated classifier was quite poor. Figure 7 shows the performance as a function of grain size (the number of concatenated messages in each virtual data set) at a dilution rate of 80 percent.

We note that as the number of messages being concatenated increases, performance drops rapidly toward chance (50 percent). If the grain size is increased further, performance remains at the chance level. In the next section we consider much larger (and more realistic) grain-size shifts of 10000. As we see from this graph, uncalibrated performance in that range would be useless.

Experiment 2: Cross training. In this experiment we use the cross-trained classifier defined in Figure 6 to classify the virtual data sets. We fix the grain size parameter G at 10000 messages in each virtual data set. This is the average size of the actual e-mail databases that we have collected—it is also large enough to allow us to explore very high dilution rates while keeping some skill-related messages in the data set. We varied the dilution rate from 90 to 99.9 percent, much higher than the 80 percent dilution rate of Experiment 1.

Figure 8 shows the results of the experiment; each point is an average of ten runs. In each run, calibration was done using 20 virtual data sets and testing on a different set of 20 such data sets. The x-axis shows the dilution rate (the percentage of skill-irrelevant messages in the positive data sets) varying nonlinearly from 100 percent down to 90 percent. The y-axis shows the percentage of data sets correctly classified.

The x-axis is displayed on a nonlinear scale in order to reveal the effects of even a vanishingly small number of skill-related messages. Note, for example, that at a dilution rate of 99.8 percent each positive data set contains *only 20* skill-related messages out of a total of 10000, yet the classifier is already scoring significantly above chance. The cross-trained classifier achieves perfect accuracy at 95 percent dilution, that is, out of 10000 messages in each data set only 500 are skill-related.

Figure 7 Uncalibrated learning performance vs grain size (number of messages in each data set). Performance tends rapidly to chance (50 percent) as grain size increases. The dilution rate (the percentage of skill-irrelevant messages in each data set) is 80 percent.

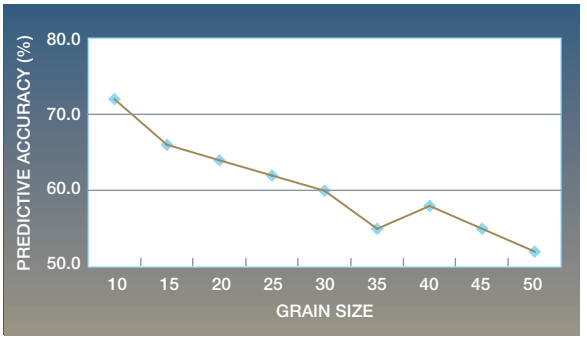
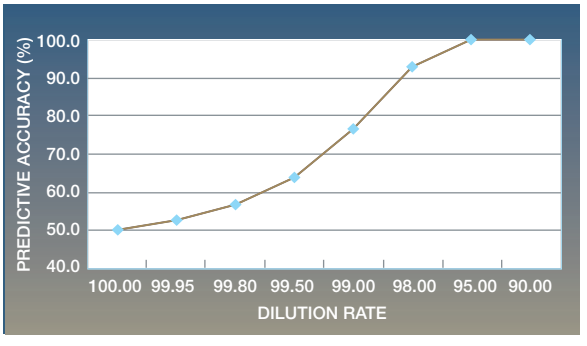


Figure 8 Cross-training performance vs dilution rate at a grain size of 10000. Note the nonlinear scale along the x-axis. The classifier does well with even an extremely small number of skill-related messages.



Analysis

The poor performance in Experiment 1 occurs because the uncalibrated classifier maximizes predictive accuracy at the message level, not the person level. It implicitly balances the cost of false positives evenly against the cost of false negatives *at the message level*. When using cross training to classify groups of messages we search for the optimal threshold; this effectively permits the cross-trained classifier to optimally balance false positives against false negatives at the person or data set level instead of the message level. This is the key to the uncalibrated classifier's difficulty. False positives are expected to be

much more expensive than false negatives in the target domain, yet the uncalibrated classifier is unaware of this and has no basis to adjust its trade-off at the message level.

To understand the asymmetry in costs between false positives and false negatives at the message level, consider that there are many messages related to any given individual. Thus the cost of misclassifying a skill-related message (that is, generating a false negative) is relatively low, since other correctly classified skill messages will still provide the evidence needed to identify possession of the skill. The number of skill-irrelevant messages for an individual, however, far outstrips the skill messages, so even low rates of false positives will be very damaging. At a 99 percent dilution of 10000 messages, even a 50 percent false-negative rate would still leave 50 correctly identified skill messages. A false positive rate of as little as 10 percent, on the other hand, will yield 1000 false-positive messages. These false positives completely overshadow the true positives, making identification impossible.

The essence of cross training is to be aware of, and compensate for, differences between the training and testing distributions.

Cross training, on the other hand, explicitly trades off false-positive and false-negative rates at both the message level and the person level. Direct comparison between the cross-trained and uncorrected classifiers is therefore in a sense unfair because the cross-trained classifier is "aware" of the relative importance of false positives at the message level. This insight highlights the essence of the cross-training methodology, namely: *being aware of, and compensating for, differences between the training and testing distributions*. Our experiments quantify the importance of explicitly making these corrections in the skill-mining domain.

To summarize the analysis, false positives are quite damaging at high dilution rates. Cross training is possible in this case because it is possible to achieve very low false-positive rates. If this were not possible it would affect the ability of cross training to compensate for large grain size and dilution rates.

Related work

It may be possible to characterize existing work as grain-size or dilution cross training, as described in this paper, but these are not the most common forms of cross training that have been reported in previous work. Below we list two very general and very common adaptations of learning that have been commonly reported in the literature.

Nonrepresentative class distributions. Examples of nontrivial cross training already exist in the machine-learning literature. One example is the *class imbalance problem*, where in the training set "one class is represented by a large number of examples while the other is represented by only a few."[9] A learning algorithm that minimizes classification error on such a domain will tend to induce classifiers that perform well on the larger classes but poorly on the smaller classes. Several techniques have been proposed to counter this problem,[9–11] most of which involve oversampling from the smaller classes or undersampling from the larger ones. This difference between the source and target domains falls into the cross-training framework.

It is interesting to note that the solution to the nonrepresentative class problem explicitly alters the false positive/negative trade-off made by the underlying classifier.[12] In our case, however, we cannot explicitly measure any shift in the relative frequency of the classes. Instead, in the skill-mining domain we compensate through the underlying asymmetrical cost of the different types of errors.

In nonrepresentative class distribution problems there is an easily determined transformation from the source to the target domain. The relative proportion of each class can be estimated for both domains and any difference can be accounted for when applying the classifier. Weiss and Provost[12] explicitly describe how to modify class probability estimates for the leaves of decision trees. This is probably the clearest existing example of cross training we have found.

Indeed we can view dilution cross training as a generalization of the nonrepresentative class distribution in which we assume that relative class probabilities have been altered by adding a previously unobserved "other" class. Unlike the nonrepresentative class distribution problem, this "other" class may asymmetrically affect the false-positive/false-negative rates for each of the originally trained

classes. This places dilution cross training between nonrepresentative class distributions and direct learning on the target distribution.

Nonstationary target distributions. Training against a nonstationary target distribution is another well-studied case where the source and target domains differ systematically. Because the target distribution is changing over time, the distribution of examples drawn previously from the source distribution is systematically distorted. This temporal information must be used to decrease the importance of older training examples. Methods that address this include windowing techniques, where only recent examples are used for training, and example-ordering techniques for order-sensitive learners such as neural networks. [13]

Future work

In the skill-mining domain, cross training can compensate for a large number of irrelevant messages and identify skills based on a relatively small number of skill messages. To understand this success, it is instructive to consider the message-level classifier induced in the first step of cross training. This classifier was able to identify many jargon terms that occur infrequently or not at all in Java-irrelevant messages (see sidebar, shown previously). Indeed, many of the terms with highest information gain are Java-related, non-English terms such as: "jdk," "jvm," "jni," "awt," "javac," and "jre." This suggests that our approach to skill mining will be successful in other cases where at least some small amount of skill-related messages are present in each person's electronic communications, provided that the skills being identified have some unique vocabulary terms.

An interesting direction for future work is to evaluate the scope of this approach by applying it to a variety of types of human skill categories and measuring its performance as a function of the jargon terms identified by the message-level classifier. Regardless of that outcome, there are a large number of technical skills that share Java programming's use of skill-specific language. Our cross-training approach is ideally suited to mining those skills.

We believe the instances of cross training addressed in this paper and previous work merely scratch the surface of interesting source/target discrepancies. Many practical learning applications also involve systematic biases on the available training data and different biases in the testing environment. We believe that developing techniques that explicitly compensate for these differences is one way to significantly improve classifier performance. An exciting direction for future work is to identify and address new instances of the cross-training framework.

Conclusions

The real world is always more complicated than our models of it. In particular, practical applications often manifest differences between the training and testing distributions, thereby falling outside the traditional classification learning framework. Motivated by such a discrepancy in the problem of automatic skill detection from e-mail, we developed the general cross-training framework. The skill detection problem, and certain other work considered previously in the literature, are instances of cross training. Although we have only begun to explore the implications of the framework, the formulations of dilution and grain-size cross training allowed us to demonstrate the efficacy of this approach for the skill detection problem, yielding good performance given only small amounts of skill-related e-mail.

Cited references

1. A. Brooking, *Intellectual Capital: Core Asset for the Third Millennium Enterprise*, International Thomson Business Press, London (1996).
2. T. H. Davenport and L. Prusak, *Working Knowledge: How Organizations Manage What They Know*, Harvard Business School Press, Boston, MA (1998).
3. T. A. Stewart, *Intellectual Capital: The New Wealth of Organizations*, Doubleday/Currency Publishers, New York (1997).
4. T. Mitchell, *Machine Learning*, McGraw-Hill Companies, New York (1997), pp. 201–202.
5. C. Faloutsos and D. Oard, *A Survey of Information Retrieval and Filtering Methods*, Technical Report CS-TR3514, Department of Computer Science, University of Maryland, College Park, MD (1995).
6. C. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*, MIT Press, Cambridge, MA (1999).
7. T. K. Landauer, P. W. Foltz, and D. Laham, "Introduction to Latent Semantic Analysis," *Discourse Processes* **25**, 259–284 (1998).
8. R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Second Edition, John Wiley & Sons, Inc., New York (2001).
9. N. Japkowicz, "Learning from Imbalanced Data Sets: A Comparison of Various Strategies," *Papers*, *AAAI Workshop on Learning from Imbalanced Data Sets*, Technical Report WS-00-05, AAAI Press, Menlo Park, CA (2000).
10. M. Pazzani, C. Merz, P. Murphy, K. Ali, T. Hume, and C. Brunk, "Reducing Misclassification Costs," *Proceedings*, *Eleventh International Conference on Machine Learning*, New Brunswick, NJ (July 10–15, 1994), pp. 217–225.
11. M. Kubat and S. Matwin, "Addressing the Curse of Imbal-

anced Data Sets: One-Sided Sampling," *Proceedings*, *Fourteenth International Conference on Machine Learning*, Nashville, TN (July 8–11, 1997), pp. 179–186.
12. G. M. Weiss and F. Provost, *The Effect of Class Distribution on Classifier Learning: An Empirical Study*, Technical Report ML-TR-44, Department of Computer Science, Rutgers University, New Brunswick, NJ (2001).
13. M. Black and R. Hickey, "Maintaining the Performance of a Learned Classifier Under Concept Drift," *Intelligent Data Analysis* **3**, No. 6, 453–474 (1999).

Accepted for publication April 25, 2002.

Daniel A. Oblinger *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (electronic mail: oblinger@us.ibm.com).* Dr. Oblinger is a research staff member at the T. J. Watson Research Center and an adjunct professor at Columbia University. He received his B.S. degree in mathematics and computer science at Northern Kentucky University, his M.S. degree in computer science at Ohio State University, and his Ph.D. degree in computer science from the University of Illinois. He has pursued his interest in machine learning, the integration of structured knowledge and learning, bioinformatics, and text mining.

Mark Reid *University of South Wales, Sidney 2052, Australia (electronic mail: mreid@cse.unsw.edu.au).* Mr. Reid is currently a Ph.D. candidate at the University of New South Wales in Sydney, Australia, where he completed his B.Sc. degree, with honors, in mathematics and computer science in 1996. His thesis is focused on the use of bias in inductive logic programming, although he also has a strong interest in reinforcement learning and computational learning theory. During the first half of 2001, he was a research intern at IBM's T. J. Watson Research Center, where he worked on a text mining application.

Mark Brodie *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598 (electronic mail: mbrodie@us.ibm.com).* Dr. Brodie is a research staff member in the Machine Learning for Systems group at the T. J. Watson Research Center and an adjunct professor at Columbia University. He did his undergraduate work at the University of Witwatersrand in South Africa. After coming to the United States, he received his Ph.D. degree in computer science in 2000 from the University of Illinois at Urbana-Champaign, working with Gerald DeJong on explanation-based learning, and has been at IBM since then. His research interests include machine learning, data mining, and intrusion detection.

Rodrigo de Salvo Braz *University of Illinois, 1304 West Springfield, Urbana, Illinois 61801 (electronic mail: braz@students.uiuc.edu).* Mr. Braz has earned B.Sc. and M.Sc. degrees in computer science from the University of São Paulo, Brazil. He is currently a computer science Ph.D. candidate at University of Illinois at Urbana-Champaign. His research focuses on probabilistic relational machine learning applied to language and vision domains.