On-line Metalearning in Changing Contexts: METAL(B) and METAL(IB)

Gerhard Widmer
Department of Medical Cybernetics and Artificial Intelligence,
University of Vienna, and
Austrian Research Institute for Artificial Intelligence,
Schottengasse 3, A-1010 Vienna, Austria
gerhard@ai.univie.ac.at

Abstract

Many real-world concepts are heavily context-dependent. Changes in context can produce more or less radical changes in the associated concepts. On-line concept learning in such domains requires the ability to recognize and adapt to such changes.

This paper concentrates on a class of learning tasks where the domain provides explicit *clues* as to the current context (e.g., attributes with characteristic values). A general two-level learning model is presented that effectively adjusts to changing contexts by trying to detect (via 'meta-learning') contextual clues and using this information to focus the learning process. Context learning and detection occur *during* regular on-line learning, without separate training phases for context recognition.

Two operational systems based on this model are presented that differ in the underlying learning algorithm and in the way they use contextual information: METAL(B) combines meta-learning with a Bayesian classifier, while METAL(IB) is based on an instance-based learning algorithm. Experiments with synthetic domains as well as a 'real-world' problem show that the algorithms are robust in a variety of dimensions, and that meta-learning produces substantial improvement over simple object-level learning in situations with changing contexts.

Motivation

The fact that concepts in the real world are not eternally fixed entities or structures, but can have a different appearance or definition or meaning in different contexts has only gradually been recognized as a relevant problem in concept learning. Michalski (1987) was one of the first to formulate it; he suggested a specialized *two-tiered representation* formalism to represent different aspects of context-dependent concepts (see also Bergadano et al., 1992). Recently, context dependence has been recognized as a problem in a number of practical machine learning projects (e.g., Katz et al., 1990; Turney, 1993; Turney & Halasz, 1993; Watrous, 1993; Watrous & Towell, 1995). There, various techniques for context normalization etc. were developed. All of these methods either assume that contextual attributes are explicitly identified by the user, or require separate pre-training phases on special data sets that are cleanly separated according to context.

We are studying the effects of context dependence and changing contexts in the framework of *incremental* (or *on-line*) learning, and we are interested in learners that can adapt to different contexts without explicit help from a teacher. The scenario is as follows: assume that a learner is learning on-line from a stream of incoming (labeled) examples. Assume further that the concepts of interest depend on some (maybe hidden) *context*, and that changes in this context can induce corresponding changes in the target concepts. As a simple example, consider weather prediction rules, which may vary drastically with the change of seasons. The visible effects of such changes are increased prediction error rates.

The development of on-line learners that can cope with concept drift and changing contexts has been the subject of recent work on the FLORA family of algorithms (Widmer & Kubat, 1993, 1996; Widmer, 1994). The basic strategy in the FLORA algorithms is to continually monitor the success of on-line prediction and to make educated guesses at the occurrence of context changes and corresponding concept changes. There is no explicit representation of contexts.

But maybe one can do better. In some domains, the data may in fact contain explicit *clues* that would allow one to identify the current context, if one knew what these clues are. Technically, such clues would be attributes or combinations of attributes whose values are characteristic of the current context; more or less systematic changes in their values might then indicate a context change.

As a simple example, consider the license plates attached to vehicles in a particular country. An agent crossing the border between, say, Austria and Germany might notice that all of a sudden the license plates look different, in a systematic way, and that might lead it to suspect that it is now in a different environment where some of the rules it had learned

before may not be valid any more. Many other examples of such *contextual clues* come to mind (climate or season in weather prediction, environmental conditions like exterior temperature in technical process control tasks, lighting conditions or background color in automatic vision, characteristics of particular rooms in in-door robotic tasks, speaker nationality and sex in speech processing, etc.). In the following, we will refer to such context-defining attributes as *contextual clues*.

In this paper, we describe a general two-level learning model, and its realization in two specific systems named METAL(B) and METAL(IB), that can *learn* to *detect* such contextual clues, and can react accordingly when a change in context is suspected. The model consists of a *base level* learner that performs the regular on-line learning and classification task, and a *meta-learner* that tries to identify attributes and features that might provide contextual clues. Context learning and detection occur *during* regular on-line learning, without separate training phases for context recognition. Perceived context changes are used to focus the on-line learner specifically on information relevant to the current context. The result is faster adaptation to changed concept definitions, and generally an increase in predictive accuracy in dynamically changing domains. At the moment, both object-level and meta-level learning are restricted to nominal (discrete) attributes, but extensions of the model to numeric domains seem rather straightforward.

Preliminaries: Bayesian Classifiers

For the moment, let us assume that our basic incremental induction algorithm is a *simple* (or *naive*) *Bayesian classifier* (as is indeed the case in the first of our algorithms to be presented below, METAL(B)). That will make it easier to explain the meta-level learning strategy, which also has a distinct Bayesian flavor.

A *simple Bayesian classifier* is a probabilistic classification scheme that uses Bayes' theorem to determine the probability that an instance belongs to a particular class, given the instance's description. In the following, we assume that examples are described in terms of (discrete) *attributes* a_i; we will use the term *feature* for a specific attribute-value combination, notated as $a_i : v_{ij}$. Examples are assumed to belong to mutually exclusive classes c_i. Bayes' theorem defines the posterior probability that some new instance I belongs to class c_i as

$$p(c_i|I) = \frac{p(c_i)p(I|c_i)}{p(I)}$$

where $p(c_i)$ is the prior probability of class c_i and $p(I|c_i)$ is the probability of an instance like I, given class c_i. Assuming that I is a conjunction of attribute values v_j, the above formula can be rewritten as

$$p(c_i|\bigwedge v_j) = \frac{p(c_i)p(\bigwedge v_j|c_i)}{\sum_k p(\bigwedge v_j|c_k)p(c_k)}$$

To make this formula operational, one usually assumes that the attributes are independent, so that $p(\bigwedge v_j|c_k)$ can be computed as the product of the $p(v_j|c_k)$.

Incremental induction of a Bayesian classifier is straightforward. One maintains a number of counters, from which the prior and conditional probabilities can be estimated: a count N of the total number of instances encountered so far, a table C_i that keeps track of the relative frequency of class c_i observed so far; a table AV_{ij} that records the number of examples with attribute value $a_i = v_{ij}$, and a table AVC_{ijk} that records the number of examples with $a_i = v_{ij}$ belonging to class c_k. Learning then simply consists in updating these counters after processing each instance. The algorithm is simple, naturally incremental, and robust in the face of noise. Its major weakness is that we must assume independence of the attributes, which severely limits the class of learnable concepts.

In the following, we take this to be our basic incremental learning algorithm, with one important modification: our learner maintains a *window* of fixed length. As new examples are added to the window, the oldest ones are deleted from it if the window size is exceeded. This is to ameliorate the problem that very old instances pertaining to an outdated context may prevent the learner from effectively adjusting to new hypotheses. The window size is a user-settable parameter, but it remains fixed during the entire learning process. The tables C_i, AV_{ij}, and AVC_{ijk} are always updated with respect to the examples in the current window.

Meta-Learning: Learning to Recognize Contextual Clues

When the underlying target concept drifts or changes due to a changed context, the Bayesian classifier (indeed, any induction algorithm that bases its hypotheses on the contents of the window) will eventually adjust to the new concept, if the new context is stable for a sufficiently long period. The smaller the window, the faster the adjustment, as old, contradictory examples will be forgotten more quickly. However, in domains that provide explicit context clues, one would expect more: the learner should learn to recognize such clues and react in some appropriate way when they signal a potential context change. To operationalize this goal, we first need to define the central notions.

Definitions

What are contextual clues? Turney (1993) was one of the first to explicitly acknowledge the problem of context in learning and to try to give a formal definition of contextual and context-sensitive features. Eventually, however, he relied on the user to explicitly identify contextual features beforehand. His particular approach was motivated by batch learning problems where the *testing examples* (i.e., those to which the learned concepts would eventually be applied) might be governed

Table 1: Tables maintained for identifying predictive and contextual attributes.

Table	Counts occurrences/rel.frequency of	Computed over	Used at
C_i	# examples in class c_i	current window	base-level
AV_{ij}	# examples with $a_i = v_{ij}$	current window	base-level
AVC_{ijk}	# examples with $a_i = v_{ij}$ in class c_k	current window	base-level
$\hat{C}_{ij}$	# examples in meta-class $\hat{c}_{ij}$	entire history	meta-level
$\hat{AV}_{ij}$	# examples with $a_i = v_{ij}$	entire history	meta-level
$\hat{AVC}_{ijkl}$	# examples with $a_i = v_{ij}$ in meta-class $\hat{c}_{kl}$	entire history	meta-level

by a different context than the training examples from which the concepts were learned. The contextual features were then used for different kinds of normalization at prediction time. In contrast, we present a method to automatically *detect* contextual attributes in an on-line learning setting and to utilize this information *during learning*.

Our operational definition of contextual attributes, i.e., attributes that provide contextual clues, is based on the notion of *predictive features*. Intuitively speaking, an attribute is predictive if there is a certain correlation between the distribution of its values and the observed class distribution. This is formalized in the following two definitions:

Definition 1 *(Predictive features).* A feature (attribute-value combination) $a_i : v_{ij}$ is *predictive* if $p(c_k \mid a_i = v_{ij})$ is significantly different from $p(c_k)$ for some class c_k.

Definition 2 *(Predictive attributes).* An attribute a_i is *predictive* if one of its values v_{ij} (i.e., some feature $a_i : v_{ij}$) is predictive.

The most obvious kind of contextual clue one could imagine is that one or more attributes would have constant values during a certain context (regardless of an instance's class). Think, for instance, of the color of the walls in a particular room. This cannot be expected in general. We will rely on a more abstract and powerful notion: a feature is considered a contextual clue if there is a strong correlation between its temporal distribution of values and the times when certain other attributes are predictive. Intuitively, a contextual attribute is one that could be used to predict which attributes are predictive at any point in time.[1] This notion is formalized in definitions 3 and 4:

Definition 3 *(Contextual features).* A feature $a_i : v_{ij}$ is *contextual* if it is predictive of predictive features, i.e., if $p(a_k : v_{kl}$ *is predictive* $\mid a_i = v_{ij})$ is significantly different from $p(a_k : v_{kl}$ *is predictive*$)$ for some feature $a_k : v_{kl}$.

[1] Indeed, contextual attributes will be used for this purpose in the algorithm METAL(IB).

Definition 4 *(Contextual attributes).* An attribute a_i is *contextual* if one of its values v_{ij} is contextual.

We thus have a two-level definition of contextual attributes, with both levels of definition being of the same type. Definition 2 (*predictive attributes*) is identical to Turney's (1993) notion of *primary feature*. Definition 4 (*contextual attributes*) is more specific and operational than Turney's definition of *contextual features* (which essentially defines an attribute as contextual if it is not in itself predictive, but would lead to less predictive accuracy if omitted). We now specify procedures to identify potentially predictive and contextual features and attributes during the incremental learning process.

Identifying contextual features through meta-learning

Assume our base-level Bayesian classifier is learning on-line from a stream of incoming examples. After each learning step, we use a *statistical χ^2 test of independence* to determine which features are currently *predictive*:

Criterion 1 *(Predictive features)*: A feature $a_i : v_{ij}$ is recognized as *predictive* if the distribution of classes in examples with $a_i = v_{ij}$ is significantly different (as determined by a χ^2 test with a given significance level) from the unconditioned distribution of classes within the current window.

Predictive features are computed relative to the *current window* because predictivity is a temporary quality that may change with time and context. The information needed for the χ^2 test is readily available in the tables C_i and AVC_{ijk} that are maintained by the base-level learner.

Contextual features are also determined by a χ^2 test, on a higher level. To this end, we define 'meta-classes' $\hat{c}_{ij}$: an instance I is in class $\hat{c}_{ij}$ if feature $a_i : v_{ij}$ is recognized as predictive at the time of classification of I. Analogously to above, tables are maintained for these meta-classes: the table $\hat{C}_{ij}$ counts the number of examples in meta-class $\hat{c}_{ij}$, $\hat{AV}_{ij}$ counts the number of examples with attribute value $a_i = v_{ij}$, seen since the

Table 2: The two-level algorithm of METAL(B)

Parameters: W (fixed window size), α (significance level for χ^2 test).

for each new incoming instance I do
begin
  CAs := current_context_attributes($\hat{C}_{ij}$, $\hat{AV}_{ij}$, $\hat{AVC}_{ijkl}$, α);
  Vs := values of attributes CAs in current instance I;
  RIs := examples in current window with values Vs for attributes CAs;
  $Class$:= class predicted for I by naive Bayesian classifier from selected examples RIs;
  add I to current window and drop oldest instance from window;
  update tables C_i, AV_{ij}, AVC_{ijk} for base-level Bayesian learning;
  PFs := currently_predictive_features(C_i, AV_{ij}, AVC_{ijk}, α);
  update tables $\hat{C}_{ij}$, $\hat{AV}_{ij}$, $\hat{AVC}_{ijkl}$ for meta-level learning, given PFs
end;

very beginning, and $\hat{AVC}_{ijkl}$ counts the number of examples with $a_i = v_{ij}$ in meta-class $\hat{c}_{kl}$. In other words, $\hat{AVC}_{ijkl}$ keeps track of the number of co-occurrences of $a_i : v_{ij}$ combinations in examples and the predictiveness of certain features $a_k : v_{kl}$. These three tables are maintained with respect to the entire learning history (not the current window), as changes of context and the emergence of skewed predictivity distributions are long-term processes. Table 1 summarizes the various tables that need to be maintained. There are then two conceivable operational criteria by which one could detect contextual features and attributes:

Criterion 2a *(Contextual features)*: A feature $a_i : v_{ij}$ is recognized as *contextual* if the distribution of meta-classes $\hat{c}_{kl}$ in examples with $a_i = v_{ij}$ is significantly different (as determined by a χ^2 test with a given significance level) from the unconditioned distribution of the $\hat{c}_{kl}$, observed over the entire learning history.

Criterion 2b *(Contextual features)*: A feature $a_i : v_{ij}$ is recognized as *contextual* if, for some feature $a_k : v_{kl}$, the distribution of meta-class $\hat{c}_{kl}$ versus $\overline{\hat{c}_{kl}}$ in examples with $a_i = v_{ij}$ is significantly different (as determined by a χ^2 test with a given significance level) from the unconditioned distribution of $\hat{c}_{kl}$ versus $\overline{\hat{c}_{kl}}$, observed over the entire learning history.

Criterion 2a pays attention to global distribution changes between the predictivity of different features, while criterion 2b is basically a direct translation of definition 3 above: $a_i : v_{ij}$ is contextual if its values correlate with the predictivity of *some* other feature $a_k : v_{kl}$. After some experimentation with both approaches, we have settled for criterion 2b (though criterion 2a yields very similar results in most cases).

Recognizing contextual features and attributes via this two-stage process constitutes an act of *meta-learning*: the base-level learning process is monitored, and the temporal coincidence of predictivity of certain features and the presence of other features in training instances leads to the identification of attributes that could provide contextual clues. The contextual features are taken as a description or *identifier* of the current context. In the following, we present two learning systems with different underlying learning algorithms that take advantage of the information provided by meta-learning.

A Bayesian Classifier with Meta-Learning: METAL(B)

Our first algorithm is called METAL(B) (META-Learning with underlying Bayes classifier) and uses a simple Bayesian classifier as its underlying ('object-level') incremental learning algorithm. It was first presented and is described in more detail in (Widmer, 1996).

In METAL(B), the contextual attributes identified by meta-learning are used to *focus* the base-level Bayesian classifier on relevant examples when making predictions: whenever a new instance comes in, the set of attributes that are currently contextual (if any) is established, and the Bayesian classifier is then made to use for prediction only those examples from the window that have the same values for the contextual attributes as the new instance to be classified. In other words, the base-level classifier uses only those instances as a basis for prediction that seem to belong to the *same context* as the incoming example. If no attributes are currently recognized as contextual, the entire window is used for Bayesian classification. After classification, the true class of the new instance is read, and the learning tables for both base and meta-level are updated. Table 2 summarizes the complete two-level algorithm of METAL(B).

A consequence of this selective strategy is that base-level prediction becomes more expensive: the Bayesian classifier can no longer use the available tables C_i, AV_{ij}, and AVC_{ijk}, as these summarize all examples

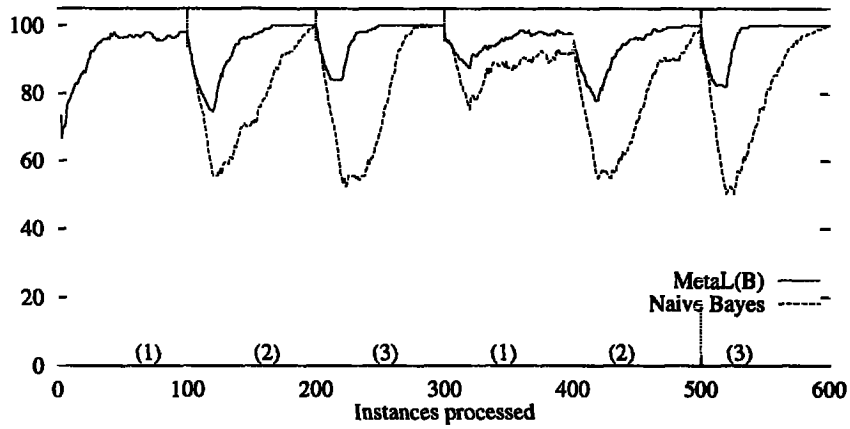

Figure 1: Effect of context identification in STAGGER concepts.

from the window. The relative frequencies required for the application of Bayes' rule must now be computed dynamically from the set of currently relevant instances (which are a subset of the window). On the other hand, this is no more expensive than the lookup in an instance-based learning algorithm (Aha et al., 1991) would be, and the fixed window size at least puts an upper bound on the number of examples that must be examined at each point in the learning process.

Note that METAL(B) depends on two *parameters*. All experiments reported below were performed with α=0.01 (99% confidence that the observed difference between conditioned and unconditioned distributions is not due to chance) and window size $W = 100$.

Experiments with METAL(B)

METAL(B)'s behavior was studied in a series of experiments in synthetic domains. Some of these are briefly described below. More detail can be found in (Widmer, 1996). In addition, METAL(B) was also evaluated on a hard 'real-world' problem where contextual effects might play a role, but the exact characteristics of the problem are not known.

Basic context identification

The first experiment demonstrates the basic effects of the context detection mechanism. The artificial test domain used here will serve as a running theme throughout most of our subsequent experiments.

METAL(B) was applied to a sequence of simple concepts that were first introduced by Schlimmer and Granger (1986) to test their concept drift tracking algorithm STAGGER. The concepts were later on also studied extensively in experiments with the FLORA algorithms (Widmer and Kubat, 1996). In a simple blocks world defined by three nominal attributes *size* $\in$ {*small*, *medium*, *large*}, *color* $\in$ {*red*, *green*, *blue*}, and *shape* $\in$ {*square*, *circular*, *triangular*}, we define a sequence of three target concepts (1) *size* = *small* $\wedge$ *color* = *red*, (2) *color* = *green* $\vee$ *shape* = *circular* and (3) *size* = (*medium* $\vee$ *large*). The (hidden) target concept will switch from one of these definitions to the next at certain points, creating situations of extreme concept drift. In addition, we introduce a fourth attribute *ctxt* $\in$ {*c*1, *c*2, *c*3}, which is used to create perfect contextual information: whenever concept (1) is in effect, all examples (positive and negative) are made to have value *ctxt* = *c*1, etc.

Figure 1 plots the on-line prediction accuracy of METAL(B) vs. the simple Bayesian classifier on the concept sequence 1-2-3-1-2-3. Sequences of 600 examples each were generated randomly and labeled according to the currently ruling concept; after every 100 instances the context plus underlying concept was made to change. On-line accuracy was computed as a running average over the last 20 predictions. The figure plots the averages over 10 (paired) runs. Parameter settings were $\alpha = 0.01$ and *window size* = 100.

The curves show convincingly that METAL(B) does make effective use of the contextual information contained in the data. We witness both a less dramatic drop in accuracy at points of context change, and significantly faster re-convergence to high accuracy levels. Obviously, METAL(B) quickly identifies the contextual attribute *ctxt* (soon after the context has first switched from 1 to 2) and from then on concentrates only on examples pertaining to the new context, whereas the naive Bayes algorithm gives equal consideration to all examples in the current window, including those the still pertain to the old context. The fact that both algorithms fail to reach an accuracy of 100% in context (1) is due to the sparsity of the concept (only 11% of the examples are positive). The improvement produced by meta-learning, however, is evident.

(Widmer, 1996) analyzes in more detail the process of identifying predictive and contextual attributes. In this particular experiment, the process works basically

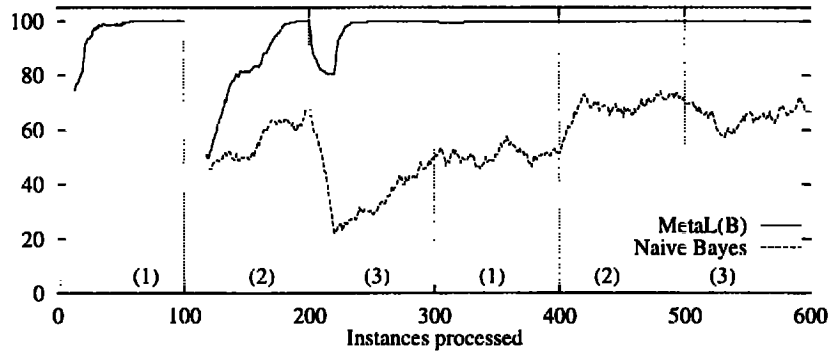

Figure 2: METAL(B) on STAGGER concepts — window size 300.

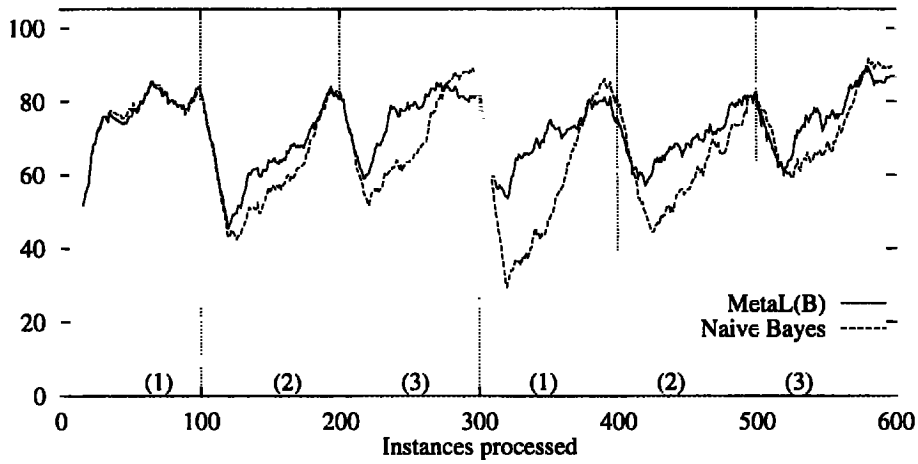

Figure 3: STAGGER concepts — 5 irrelevant attributes and 20% attribute noise.

as expected: the attributes *color* and *size* are recognized as predictive about midway during context (1), *color* and *shape* during context (2), and *size* during context (3). The attribute *ctxt* is recognized as a contextual clue towards the beginning of the (first occurrence of) context (2), due to that fact that *size* ceases to be predictive, and *shape* becomes predictive.

The effect of the system parameters

METAL(B) depends on two parameters: the significance level α used in the χ^2 tests at two levels, and the fixed *window size*. A level of $\alpha = 0.01$ has produced good results in all our experiments. Less strict significance levels make the meta-learner less discriminating: it becomes more easily possible for features to be accidentally 'recognized' as predictive for short periods, which causes frequent changes in the distribution of predictive features. The effect is instability. On the other hand, tighter significance levels (e.g., $\alpha = 0.001$) have left our results virtually unchanged.

As for the *window size*, the smaller the window, the faster the base-level learner will adapt to changes, and thus the smaller the advantage afforded by meta-learning. Too narrow a window is detrimental, as the Bayesian classifier bases its predictions on too few data points. Too large a window, on the other hand, reduces the base-level learner's ability to adjust to changes and, if it permanently contains conflicting instances from different contexts, may prevent the learner from ever reaching high predictive accuracy. Figure 2 plots the results on the STAGGER task when the window size is set to 300. METAL(B) detects the contextual attribute somewhere during context (2) and uses its values to always select the correct examples for prediction. After the first 300 examples, the window always contains instances from all three contexts, and METAL(B) can perfectly predict from then on. For the same reason, the simple Bayesian classifier fails completely.

Irrelevant attributes and noise

Bayesian learners are known to be quite robust in the face of irrelevant attributes and noise. Experiments

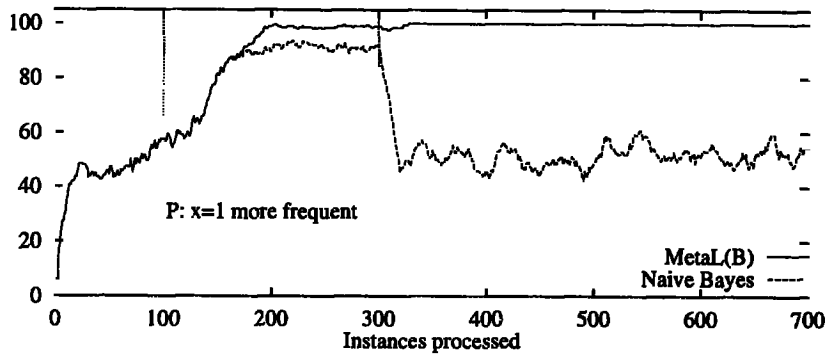

Figure 4: "Quasi-contextual" learning: XOR with 5 irrelevant attributes.

with METAL(B) have confirmed this, both for object-level and meta-level learning. As a rather extreme example, Figure 3 plots the performance on the STAGGER task when the examples were subjected to 20% attribute noise,[2] with an additional 5 irrelevant attributes with random values. It is evident that meta-learning still leads to improved adjustment to changes, though such high noise levels may eventually produce effects of instability, as we can see in context (3). Table 5 below gives more information.

Other experimental dimensions

Gradual vs. abrupt context changes:
Experiments also confirm that METAL(B) is rather insensitive to the *speed* of a context change, i.e., whether a new context takes over abruptly or only gradually. Unlike systems like FLORA4 (Widmer, 1994), METAL(B) has no problems with slow drift — it is the overall distributions of the predictivity of features that determine METAL(B)'s behavior, not necessarily dramatic short-term changes. This is important, because many realistic applications are presumably characterized by more or less gradual (and noisy) context changes. More details can be found in (Widmer, 1996).

Complex and imperfect contextual clues:
In real life, contexts may be defined by more complex combinations of features, and contextual attributes may be changing. Preliminary experiments suggest that METAL(B) has no problems with *conjunctively* defined contexts. Contexts defined by *disjunctions* of features create problems, especially when the contextual attributes are not the same in different contexts. The main problem is that contextual information is used conjunctively in METAL(B)'s focusing strategy, which makes the base-level learner rely on very few examples in some cases. Generally, of course, the meta-learner suffers from the same limitation as the base-level Bayesian classifier: it must assume independence between *contextual attributes*.

Another potential complication is that contexts may not always be characterized by perfectly constant values. Experiments with noise in the context attributes suggest that METAL(B) is robust to that kind of imperfection, but more refined experiments will be needed to study other types of imperfect clues (e.g., changing value distributions).

"Quasi-contextual learning":
An interesting side-effect of meta-learning is that it may in certain cases help the Bayesian learner to overcome its inherent representational limitations, incurred by the attribute independence assumption. As an example, consider the XOR function: in the concept $x = 1 \oplus y = 1$, neither of the two attributes x and y in isolation contributes directly to the classification. A Bayesian classifier will always linger around the base-level accuracy of 50%, given a set of uniformly distributed examples. The same holds for METAL(B), as long as the examples are presented in random order. However, if for some time examples appear in a skewed distribution, meta-learning may exploit this by learning to regard one of the attributes as contextual and the other as predictive. This two-level view of the XOR function would then allow the system to perfectly classify from that point on: if context is $x = 1$, then $y = 0$ implies XOR, and vice versa.

Figure 4 shows the learners on sequences of XOR examples with five additional irrelevant attributes, where during a certain period P (between the 100th and the 300th instance), examples — both positive and negative — with $x = 1$ were more frequent than those with $x = 0$ (90% vs. 10%). Before example 100 and after example 300, instances were presented in a uniform distribution. The results are again averages over 10 runs.

The effect is very clear: METAL(B) does indeed single out x as the contextual attribute at some point dur-

[2] A noise level of η% means that for each attribute in a training example, its true value is replaced by a random value from its domain with probability $\eta/100$. Both the 'base-level' attributes *color*, *size*, and *shape* and the context attribute *ctxt* were equally subjected to noise.

Table 3: Results of Schubert experiment.

Learning algorithm	Mean acc. (%)	Std. dev.	# runs better	max.better
Naive Bayes:	68.75	1.07	0	---
METAL(B):	74.62	1.44	50	9.22

ing period P, which allows it to reach an accuracy level of (almost) 100% quickly, and also to hold on to this level during the following random period. The simple Bayesian classifier improves its performance between examples 100 and 300 (it can never reach 100%), but quickly drops to 50% as the instances are again uniformly distributed.[3] We may conclude from this that meta-learning can be useful also in settings that are not normally thought of as characterized by changing contexts.

Learning from real data

METAL(B) was also tested on a difficult 'real-world' problem with unknown characteristics. The problem comes from the domain of tonal music and consists in learning to predict (on-line) what chord should accompany the next note in a given melody. Specifically, the task is to correctly predict one of three classes: *tonic harmony* (e.g., the note is to be played with a C major chord, if the piece is in the key of C major), *dominant* (i.e., a G major chord in the key of C), or *other*. In terms of a real scenario, imagine a guitar player who is trying to accompany a singer in real time on pieces she does not know and tries to get at least the two most important chord types (tonic and dominant) right.

The data used for the experiment were the melodies of Franz Schubert's *German Mass*, a collection of 8 songs of varying length (between 42 and 113 notes). There are 553 melody notes in total. The distribution of classes is 290 (52%) *tonic*, 179 (32%) *dominant*, and 84 (15%) *other*.

The individual notes were described by 11 discrete attributes: the *mode* of the current piece (major or minor), the *meter* (e.g., 4/4, 3/4, or 6/8), the current *tactus* (i.e., whether the major metrical level — the level at which one would tap one's foot in rhythm — is the level of quarter of eighth notes), the current *local key* (to describe modulations within a piece), and various attributes that describe the current note itself and its predecessor: *scale degree* (a tonality-independent abstraction of the note name), *duration*, and *metrical strength* of the current note, *duration* of the note's predecessor, the *interval* and its *direction* between the previous and the current note, and the *harmony* that accompanied the previous note.

[3]That METAL(B) fails to achieve a perfect 100% during period P, but then performs perfectly afterwards may seem surprising. The explanation lies in the highly unbalanced instance distribution during P. See (Widmer, 1996) for more detail.

We conjectured that more global properties like mode, meter, tactus, and local key might have a context-defining effect in certain cases, i.e., that the rules determining the harmonic role of a note might be slightly different in some of these contexts. However, we don't know this in detail, and the contextual effects, if any, might be weak and difficult to discover.

What makes the problem even harder is that the given attributes are highly *inadequate*: there are numerous cases of notes with the same description but different classification. Harmonic choices are by no means unique, and the specific decision also depends on aspects of larger context (musical form, harmonic rhythm, etc.) that are not captured by our local representation. It is thus clearly impossible to achieve a predictive accuracy of anything close to 100%.

To reduce the effect of the specific ordering of the songs, the algorithms were run on 50 random permutations of the 8 songs. Table 3 shows the results in terms of total classification accuracy. METAL(B)'s advantage of 5.87 percentage points is significant at the 0.001 level, according to a two-sided t-test. It scored better than the simple Bayesian classifier in *all* of the 50 runs, with a maximum advantage of 9.2 percentage points.

The attributes most often singled out as contextual were meter and tactus, less frequently mode, and very rarely local key (which was surprising to us, but probably indicates that the periods of modulation are too short and unstable to be contextually significant). Interestingly, also note duration was sometimes considered contextual: it does not help in directly predicting the harmony, but it is useful as a 'secondary' decision criterion. In other words, there is some dependency between this attribute and some more predictive ones, and METAL(B) resolves the dependency by treating note duration as a contextual attribute.

As a kind of afterthought, we propose an alternative view of meta-learning: instead of a focusing or selection strategy, it could also be interpreted as a process of *transfer* of (learned) knowledge *across contexts*. That leads one to ask the following question: could it be that the 8 pieces are so different that there cannot be much useful transfer from one piece to the next, in other words, that one would achieve better results by learning from each piece *separately*, simply starting from scratch with every new piece? And indeed, it turns out that the base-level learner, when run on each piece separately, reaches a total accuracy over the 8 songs of 69.54%, which is slightly *higher* (though not at a sta-

tistically significant level) than the 68.75% achieved by simple Bayesian learning with a fixed window over the whole sequence! METAL(B), on the other hand, achieves markedly higher accuracy. The conclusion is that indiscriminate transfer can indeed be harmful, but METAL(B) performs what might be called *selective cross-contextual transfer* — only those pieces of information are carried over that appear relevant to the current context.

An Alternative Realization of the Model: METAL(IB)

METAL(B) is but one possible incarnation of a very general learning framework. It seemed natural and elegant to use a Bayesian classifier for object-level learning, because metalearning as defined here also has a distinct Bayesian flavor. However, it should be possible in principle to use any other incremental induction algorithm for base-level learning. Given the serious inherent limitations of a naive Bayesian classifier, we have recently developed an alternative realization of the general model: METAL(IB) (METALearning with underlying Instance-Based classifier) realizes the same meta-learning strategy as METAL(B), but uses a simple *instance-based* learning algorithm (Aha et al., 1991) as base-level learner. New incoming examples are classified by a straightforward 1-NN (nearest-neighbor) method, using Euclidean distance as the (dis)similarity measure. Basis for prediction are the examples in the current window.

An instance-based learner allows us to use context information in a more flexible way, e.g., for feature weighting, which is not directly possible in METAL(B)'s Bayesian classifier. Also, instance-based learners can in principle approximate any target function, while Bayesian classifiers are severely limited. In the following, we present four variants of METAL(IB) that differ in the way they use the information about suspected contextual clues; their advantages and shortcomings, especially in relation to METAL(B), are then briefly investigated in the following section:

1. **Exemplar Selection** — METAL(IB)–ES:
   Here, contextual information is used in the same way as in METAL(B), namely, to *select relevant exemplars* during classification: only those instances from the window are used for prediction that have the same values for the current context attributes (i.e., presumably belong to the same context) as the current instance (the one to be classified).
2. **Exemplar Weighting** — METAL(IB)–EW:
   In this variant, contextual clues are used for exemplar *weighting* rather than strict selection. When classifying by nearest neighbor, each instance (*exemplar*) in the current window is assigned a weight W, and the measured similarity between new instance and exemplar is multiplied by W. The idea is that exemplars that are more likely to belong to the same context as the new instance should have more influence in classification. The weight is computed as

$$W(E) = 1/(1 + D_{|CAs|}(E, CI))$$

   where $D_{|CAs|}$ is the distance between the exemplar E and the current instance CI, measured only with respect to the context attributes CAs.
3. **Feature Weighting** — METAL(IB)–FW:
   Here, instead of weighting the examples in the window, METAL(IB) uses a weighted similarity measure that assigns different importance to individual attributes during nearest neighbor classification. Features or attributes believed to be *predictive relative to the current context* should receive correspondingly higher weights. To this end, we augment the meta-level algorithm with a component that *predicts* which attributes are or should be *predictive* for each incoming example, *given the instance's values for the currently recognized context attributes*. This can be easily done by using Bayes' rule on the tables updated in the meta-learning process ($\hat{C}_{ij}$, $\hat{AV}_{ij}$, and $\hat{AVC}_{ijkl}$). METAL(IB) computes a weight for each attribute as follows:

$$W(A) = 1 + P_{pred}(A)$$

   where $P_{pred}(A)$ is the probability which the system assigns to the belief that A is a predictive attribute relative to the current instance (computed via Bayes' rule at the meta-level).
4. **Combined Exemplar/Feature Weighting** — METAL(IB)–COM:
   This variant performs both exemplar and feature weighting.

Our expectations were that weighting approaches would generally be less brittle than strict exemplar selection, and that *feature weighting* in particular would produce a new, interesting effect: as the feature weights are derived, via Bayes' rule, from METAL(IB)'s meta-level tables ($\hat{C}_{ij}$, $\hat{AV}_{ij}$, $\hat{AVC}_{ijkl}$), which in turn are a kind of *memory of past contexts*, this feature weighting strategy should enable the system to readjust more quickly to contexts that have already occurred before.

METAL(IB)–COM, finally, should combine the advantages of exemplar weighting (fast reaction to perceived context change by disregarding exemplars in the window that pertain to an outdated context) and feature weighting (fast adjustment to contexts that have been encountered before).

Preliminary Experimental Results

Here are some preliminary results. Figure 5 compares the simple instance-based learner (with fixed window of size 100) with its various meta-learning

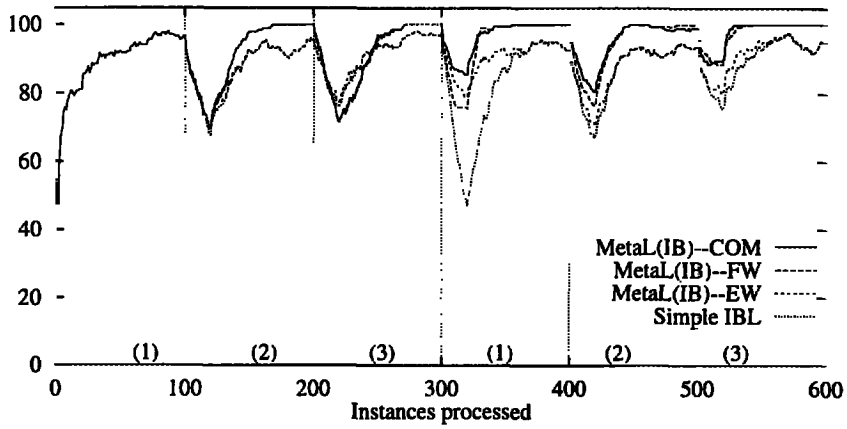

Figure 5: Context identification in METAL(IB).

Table 4: METAL(B) vs. METAL(IB) on simple STAGGER problem.

	No irrelevant attrs Acc.% (σ)	10 irrelevant attrs Acc.% (σ)
Naive Bayes:	82.75 (2.02)	77.51 (2.12)
METAL(B):	95.75 (0.49)	85.90 (3.07)
Simple IBL:	88.10 (0.69)	68.68 (2.34)
METAL(IB)-ES:	90.05 (0.91)	74.35 (3.48)
METAL(IB)-EW:	90.05 (0.91)	74.38 (3.42)
METAL(IB)-FW:	93.75 (0.73)	73.70 (2.57)
METAL(IB)-COM:	94.23 (0.69)	78.03 (4.09)

variants[4] on the basic STAGGER task (compare this to figure 1). Again, the improvement brought about by meta-learning is evident. Among the meta-learning approaches, feature weighting performs markedly better than exemplar weighting (and equally exemplar selection), and the two feature weighting variants (METAL(IB)-FW and METAL(IB)-COM) do indeed seem to adjust to contexts faster when they reappear (see the second appearance of contexts (1), (2), and (3)). However, this claim has yet to be tested in more extensive experiments.

Table 4 summarizes the results of this experiment, as well as another experiment with 10 additional irrelevant attributes, in terms of the *total predictive accuracy* achieved by each learner over the entire sequence of 600 training instance (averaged over the 10 runs, σ = standard deviation), and also gives the corresponding figures for METAL(B). The figures establish METAL(IB)-COM as the best of the METAL(IB) strategies. We note that simple instance-based learning is markedly better than simple Bayesian classification in the task with no irrelevant attributes. We may attribute this to the inherent representational limitations of Bayesian classifiers; METAL(B)'s result shows that this handicap is compensated by meta-learning. In contrast, the instance-based algorithms are significantly inferior in the situation with 10 irrelevant attributes, which confirms one of the fundamental (and well-known) shortcomings of instance-based approaches.

A similar picture emerges when the data are subjected to *attribute noise*. Table 5 lists overall accuracies in the STAGGER domain (with 5 additional irrelevant attributes), for different noise levels. Again, meta-learning brings considerable improvement, but the amount of improvement decreases with higher noise levels. The combined strategy METAL(IB)-COM again turns out to be the best METAL(IB) variant, but the Bayesian system METAL(B) clearly outperforms the instance-based learners, indicating that both noise and irrelevant attributes are detrimental to instance-based approaches.

A domain where METAL(IB) turns out to be superior to METAL(B) is our musical chord prediction problem, where simple instance-based learning achieved an average total accuracy of 76.14% and METAL(IB)-COM reached 79.58% (compare this to 68.75% and 74.62%, respectively, for the Bayesian

[4]The curve of METAL(IB)-ES is not included in this plot; exemplar selection and exemplar weighting performed nearly identically in all our experiments.

Table 5: Overall results on noisy STAGGER problem (with 5 irrelevant attributes).

	0% noise Acc.% (σ)	10% noise Acc.% (σ)	20% noise Acc.% (σ)	40% noise Acc.% (σ)
Naive Bayes:	79.70 (1.20)	75.73 (1.96)	73.78 (1.41)	66.80 (1.93)
METAL(B):	88.40 (2.41)	80.95 (1.79)	75.60 (2.48)	66.00 (2.22)
Simple IBL:	72.40 (1.81)	68.03 (2.17)	63.78 (2.02)	60.05 (1.30)
METAL(IB)-ES:	78.03 (2.53)	72.10 (2.96)	66.00 (2.45)	61.11 (1.46)
METAL(IB)-EW:	77.95 (2.45)	72.12 (2.88)	65.90 (2.41)	61.11 (1.50)
METAL(IB)-FW:	80.47 (1.76)	73.23 (2.57)	67.80 (2.00)	61.37 (2.34)
METAL(IB)-COM:	83.22 (2.12)	75.57 (2.87)	68.58 (2.41)	62.62 (2.22)

learners — see table 3). Here, the representational flexibility of an instance-based approach seems to pay off.

There are a number of other known methods in instance-based learning for exemplar selection, exemplar weighting, and feature weighting (see, e.g., Aha, 1989; Aha et al., 1991; Cost & Salzberg, 1993) that should be experimentally compared to METAL(IB). Preliminary results achieved with David Aha's algorithms *IB3* (which performs a form of dynamic exemplar selection based on monitored predictive accuracy) and *IB4* (dynamic feature weighting) in the musical chord prediction problem suggest that METAL(IB) is clearly superior on this problem (79.6% for METAL(IB)-COM vs. 61.1% (IB3) and 68.3% (IB4)). More extensive and detailed experimental comparisons are under way.

Conclusion

The main contribution of this paper is to have shown that it is indeed possible for an incremental learner to autonomously detect, during on-line object-level learning, contextual clues in the data if such exist. The key is an operational definition of predictive and, based on those, contextual features. Identification and subsequent use of contextual information is an act of *meta-learning*. There are various ways in which such context information can be used. This paper has presented two different realizations of the meta-learning model: METAL(B) relies on a Bayesian classifier as the underlying incremental learner and uses context information to select those known instances as a basis for prediction that seem to belong to the same context as the new instance. METAL(IB) is based on an instance-based algorithm and uses contextual information for exemplar and feature weighting. The feasibility of context recognition and its benefits have been shown in a number of experiments.

Both METAL(B) and METAL(IB) are currently limited to domains described by symbolic, discrete attributes. A generalization to numeric features should not prove too difficult. Instead of maintaining counts of attribute-value and attribute-value-class combinations, the Bayesian classifier could simply assume that numeric features follow a normal distribution and keep track of the mean values and standard deviations. At the meta-level, the χ^2 tests would have to be replaced by an appropriate test of independence for continuous distributions.

Generally, we regard the work presented here as a small first step into what might become a rich field of research. The identification of contextual features is a first step towards *naming*, and thus being able to *reason about*, contexts. That is the level where we expect the full power of meta-learning to become apparent. Reasoning and learning about contexts could be used in a variety of ways and for a number of purposes, e.g., constructive induction, the recognition of (and faster readjustment to) previously encountered contexts, the emergence of expectations and predictions of the next context, etc.

There is a very interesting connection between our learning model and the notions of *transfer* and *life-long learning*, as recently proposed by Thrun and Mitchell (1995). As noted above in the context of the Schubert experiment, our learners can be interpreted as performing *cross-contextual transfer*, and they certainly are 'lifelong' learners. At first sight, the two models might appear to be orthogonal (one performs transfer across learning tasks, the other across contexts within a single task), but there are interesting parallels, and further research might lead to the formulation of a more general and powerful model that integrates both aspects of transfer.

Acknowledgments

The Austrian Research Institute for Artificial Intelligence gratefully acknowledges financial support from the Austrian Federal Ministry for Science, Research, and the Arts.

References

Aha, D.; Kibler, D.; and Albert, M. 1991. Instance-based learning algorithms. *Machine Learning* 6(1):37–66.

Aha, D. 1989. Incremental, instance-based learning of independent and graded concept descriptions.

In *Proceedings of the 6th International Workshop on Machine Learning (ML-89)*. San Mateo, CA: Morgan Kaufmann.

Bergadano, F.; Matwin, S.; Michalski, R.; and Zhang, J. 1992. Learning two-tiered descriptions of flexible contexts: The POSEIDON system. *Machine Learning* 8(1):5–43.

Cost, S., and Salzberg, S. 1993. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* 10(1):57–78.

Katz, A.; Gately, M.; and Collins, D. 1990. Robust classifiers without robust features. *Neural Computation* 2(4):472–479.

Michalski, R. 1987. How to learn imprecise concepts: A method employing a two-tiered knowledge representation for learning. In *Proceedings of the 4th International Workshop on Machine Learning*. Los Altos, CA: Morgan Kaufmann.

Schlimmer, J., and Granger, R. 1986a. Beyond incremental processing: Tracking concept drift. In *Proceedings of the 5th National Conference on Artificial Intelligence (AAAI-86)*. Los Altos, CA: Morgan Kaufmann.

Schlimmer, J., and Granger, R. 1986b. Incremental learning from noisy data. *Machine Learning* 1(3):317–354.

Thrun, S., and Mitchell, T. 1995. Learning one more thing. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*. San Mateo, CA: Morgan Kaufmann.

Turney, P., and Halasz, M. 1993. Contextual normalization applied to aircraft gas turbine engine diagnosis. *Journal of Applied Intelligence* 3:109–129.

Turney, P. 1993. Robust classification with context-sensitive features. In *Proceedings of the 6th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-93)*. Edinburgh, Scotland.

Watrous, R., and Towell, G. 1995. A patient-adaptive neural network ecg patient monitoring algorithm. In *Proceedings Computers in Cardiology 1995*. Vienna, Austria.

Watrous, R. 1993. Speaker normalization and adaptation using second-order connectionist networks. *IEEE Transactions on Neural Networks* 4(1):21–30.

Widmer, G., and Kubat, M. 1993. Effective learning in dynamic environments by explicit context tracking. In *Proceedings of the 6th European Conference on Machine Learning (ECML-93)*. Berlin: Springer Verlag.

Widmer, G., and Kubat, M. 1996. Learning in the presence of concept drift and hidden contexts. *Machine Learning* (in press).

Widmer, G. 1994. Combining robustness and flexibility in learning drifting concepts. In *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI-94)*. Chichester, UK: Wiley & Sons.

Widmer, G. 1996. Recognition and exploitation of contextual clues via incremental meta-learning. In *Proceedings of the 13th International Conference on Machine Learning (ICML-96)*. San Francisco: CA: Morgan Kaufmann. Extended version available as Report TR-96-01, Austrian Research Institute for Artificial Intelligence, Vienna.