An Inductive Logic Programming Approach to the Classification of Phases in Business Cycles

Katharina Morik, Stefan Rüping
Univ. Dortmund, Computer Science Department, LS VIII
morik@ls8.informatik.uni-dortmund.de

Abstract A workbench for knowledge acquisition and data analysis is presented and its use for the classification of business cycles is investigated. Inductive Logic Programming (ILP) allows to model relations between intervals, e.g. time or value intervals. Moreover, the user of the workbench is supported in inspecting the learned rules, not only with respect to their coverage, accuracy, and redundancy, but also regarding consistency (i.e., logical contradictions). The application of ILP requires pre-processing in order to establish time and value intervals. To this end, top-down induction of decision trees is used. This paper describes the workbench MOBAL, its learning algorithm RDT, the pre-processing of data, and the first encouraging results on business cycle data from Germany.

1 Introduction

The observation of ups and downs of business activities has been observed since a long time [1]. It is, however, hard to capture the phenomenon by a clear definition. The National Bureau of Economic Research (NBER) defines business cycles as "*recurrent sequences of altering phases of expansion and contraction in the levels of a large number of economic and financial time series.*" This definition points at the multi-variate nature of business cycles. It does not specify many of the modeling decisions to be made. There is still room for a variety of concepts.

- What are the indices that form a phase of the cycle? Production, employment, sales, personal income, and transfer payments are valuable indicators for cyclic economic behavior. Are there others that should be included?
- Which measurements of indices are to be taken? Where the classical business cycle is expressed according to the level of indicators, the growth cycle is measured with respect to the deviation from the trend of indicators.
- What is the appropriate number of phases in a cycle? The number of phases in a cycle varies in economic models from two to nine. The NBER model indicates two alternating phases. The transition from one phase to the next is given by the turning points *trough* and *peak*. In the RWI model, a cycle consists of a lower turning point, an upswing, an upper

[1] Amstad reports the first definition from Clement Juglar in 1860 [2]. She investigates several models of the business cycle and discusses their distinctions with respect to dating turning points of the business cycle.

turning point, and a downswing. Here, the turning points are phases that cover several months.

- Are all cycles following the same underlying rules or has there been a drift of the rules? What is the appropriate sample for classifying current business data?

All modeling decisions are to be (comparatively) validated with respect to economic theory and to business data. One approach to validation is the formalization by macro-economic equations. A model of business cycles is calculated *ex post* and the deviation of the results of the equations from the observed values assesses the model. For instance, the business cycle model of the Rheinisch-Westfälisches Institut für Wirtschaftsforschung (RWI) only deviated 1.2 per cent for the spring 2000 state of affairs in Germany [5]. The main focus here lies on the prediction of level or growth of business activities. We do not contribute to this approach. The other approach is an empirical one, in which statistical methods are adjusted to business data and used for prognoses. Again, the validity of statistical models is validated on past data. We are concerned with the development and comparison of methods for the empirical modeling of business cycles. Empirical methods are particularly demanded for the task of dating turning points or phases of the business cycle. Our question is: Which methods can support modeling and validating models of the business cycle? More precisely: Can inductive logic programming support economists in dating and predicting phases of the business cycle? We may re-formulate the questions into two general problems.

Dating: Given current (and past) business measurements, in which phase is the economy currently? In other words, the current measurements are to be classified into the phases of a business cycle.

Prediction: Given current (and past) business measurements, what do we expect next?

Linear discriminant analysis has been proposed as the baseline of empirical models [**?**]. Univariate rules were learned that used threshhold values for separating phases. The accuracy of the 18 learned rules was 54% in cross validation. Using this result as the baseline means that the success of any other method has to be shown in comparison to this accuracy. It has been investigated how the classification can be enhanced by the use of monthly data [4]. More sophisticated statistical models have been developed and achieved 63% accuracy [**?**]. However, even this substantial enhancement still reflects how hard it is to classify business phases correctly.

The difficulty of the problem lies in its multi-variate nature, which follows from the definition of business cycles. Moreover, the business cycle cannot be observed directly and main factors of influence may well be hidden. Hence, we may want to incorporate economic knowledge (theory) into business cycle data analysis. In fact, the advanced Markov Switching model as it was used by Sondhauss and Weihs expresses knowledge about the past phase and the transition probability to the next phase [**?**]. Also the approach which we describe in this paper, exploits domain knowledge. Here, economic knowledge is used to restrict the space of possible rules in order to exclude those that would not make sense or are trivially true.

In this paper, we investigate the applicability of inductive logic programming to the problem of dating phases of a business cycle. We were given quarterly data for 13 indicators concerning the German business cycle from 1955 to 1994, where each quarter had been classified as being a member of one of four phases [3]. The indicators are:

IE	real investment in equipment (growth rate)
C	real private consumption (growth rate)
Y	real gross national product (growth rate)
PC	consumer price index (growth rate)
PYD	real gross national product deflator (growth rate)
IC	real investment in construction (growth rate)
LC	unit labour cost (growth rate)
L	wage and salary earners (growth rate)
Mon1	money supply M1
RLD	real long term interest rate
RS	nominal short term interest rate
GD	government deficit
X	net exports

We experimented with different discretizations of the indicator values. The discretization into ranges (levels) of values was also used in order to form time intervals. A sequence of measurements within the same range is summarized into a time interval. For instance, the money supply being *high* for quarters 8 to 18 is summarized by the fact `mon1(i1,high)`, where `i1` corresponds to the time intervall from 8 to 18. Hence, the time intervals differ from indicator to indicator. Relations between the different time intervals express precedence or domination of one indicator's level to another ones level. We also compared the two phase with the four phase business cycle. In summary, the following three models were inspected:

- business cycle with four phases, without time intervals,
- business cycle with four phases, time intervals,
- business cycle with two phases, without time intervals.

Particular attention was directed towards the appropriate sample size for the dating problem. The homogenity of the data set of business cycles with two phases was investigated. The hypothesis being that at the end of cycle 3 (i.e., third quarter of 1971) the rules for dating phases could change.

2 Inductive Logic Programming

Inductive Logic Programming (ILP) establishes the intersection of logic programming and machine learning. A logic program is learned from observations by inductive reasoning. The logic program expresses a theory in the form of facts and rules in a restricted first-order logic. The theory describes diverse concepts together with the relations among them. This contrasts with propositional logic, where only one concepts and its sub-concepts can be modeled. A simple example illustrates this.

Given the observations

$mother$	$(ann,$	$brigid).$	$mother$	$(alice,$	$bonnie)$
$mother$	$(brigid,$	$cecilie).$	$mother$	$(bonnie,$	$christie)$
old	$(ann).$		old	$(alice).$	
$grandmother$	$(ann,$	$cecilie).$	$grandmother$	$(alice,$	$christie).$

ILP may learn the rule

$mother(X,Z), mother(Z,Y) \rightarrow grandmother(X,Y)$

Whereas a propositional learner cannot exploit the relations but can only learn the heuristic:

$old(X) \rightarrow grandmother(X,Y)$

A logic program is directly executable. The learned rules derive the conclusion from new facts. For instance, the learned grandmother rule derives

$grandmother(agatha, carol).$

as soon as the facts are stated:

$mother(agatha, beth).\ mother(beth, carol).$

The expressive power of first-order logic proves especially successful in relations between intervals. Explicitly the starting and end point of an interval can be stated together with the relations between intervals. For instance, direct precedence can easily be expressed between time intervals, here between the time intervals from $T1$ to $T2$, from $T2$ to $T3$, and from $T3$ to $T4$:

$cooking(C,T1,T2), serving(S,T2,T3), eating(E,T3,T4)$ (direct precedence)

Similarly, inclusion and overlap of intervals is written.

$cooking(C,T1,T4), serving(S,T2,T3), T1 \leq T2, T3 \leq T4$ (inclusion)

$cooking(C,T1,T3), serving(S,T2,T4), T1 \leq T2, T3 \leq T4$ (overlap)

It has been shown that the time relations of Allen's calculus [1] can be expressed in the form of a logic program [12, 13].

Note, that the first-order logic rules are inherently multi-variate. Distinct events can be expressed with their properties. For instance, the activities cooking (C), serving (S), and eating (E) can be described independently from another, naming the involved places ($Y1, Y2$), persons $X1$ to $X5$, and their properties ($salary$) and relations ($mother, father$).

$cook$	$(X1,$	$C).$	$guest$	$(X3,$	$E).$
$salary$	$(X1,$	$W1).$	$guest$	$(X4,$	$E).$
$kitchen$	$(Y1,$	$C).$	$guest$	$(X5,$	$E).$
$recipe$	$(Z,$	$C).$	$mother$	$(X3,$	$X5).$
$waiter$	$(X2,$	$S).$	$father$	$(X4,$	$X5).$
$salary$	$(X2,$	$W2).$			
$diningRoom$	$(Y2,$	$S).$			

Of course, this representation can be compiled down to propositional logic, if the number of objects is finite [9]. However, the ease of understanding is lost in the compilation. The understandibility of first-order logic eases the formulation of hypotheses that the learning algorithm should test on the data. We shall see, how user-specified sets of hypotheses are represented and tested by the Rule Discovery Tool.

A last advantage of ILP to be mentioned is the explicit statement of background knowledge, commonly in terms of facts. In the grandmother example, the facts stating the grandmother role are the examples and the facts stating the mother role are the background knowledge.

We may now state the task of concept learning within ILP formally.

Concept learning or learning classifications

Given positive and negative examples $\mathcal{E} = \mathcal{E}^+ \cup \mathcal{E}^-$ in a representation language $\mathcal{L}_\mathcal{E}$ and background knowledge $\mathcal{B}$ in a representation language $\mathcal{L}_\mathcal{B}$,
find a hypothesis $\mathcal{H}$ in a representation language $\mathcal{L}_\mathcal{H}$, which is a (restricted) first-order logic, such that

(1) $\mathcal{B}, \mathcal{H}, \mathcal{E}^+ \not\models \Box$ (consistency)

(2) $\mathcal{B}, \mathcal{H} \models \mathcal{E}^+$ (completeness of $\mathcal{H}$)

(3) $\forall e^- \in \mathcal{E}^- : \mathcal{B}, \mathcal{H} \not\models e^-$ (accuracy of $\mathcal{H}$)

2.1 MOBAL

MOBAL is a workbench which allows users to easily enter facts and rules, detects inconsistencies in the knowledge base, and proposes minimal changes to facts and rules in order to make it consistent [10]. In addition to the support of users in building up a knowledge base, the rule discovery tool automatically learns rules from facts and adds the learned rules to the knowledge base.

2.1.1 The Rule Discovery Tool RDT

For learning rules from facts, the Rule Discovery Tool RDT forms all possible rules according to a user given hypothesis space [6]. The user specifies rule schemata. A rule schema has predicate variables that can be instantiated by predicates of the domain. An instantiated rule schema is a rule. Rule schemata are partially ordered according to their generality. For our learning task of dating business data according to the business cycle, we first used the following rule schemata:

m1 (Index1, Value, Phase):
$Index1(T, V), Value(V) \rightarrow Phase(T)$

m2 (Index1,Value,Index2, Phase):
$Index1(T, V), Value(V), Index2(T, V) \rightarrow Phase(T)$

m3 (Index1, Value1,Index2,Value2,Phase):
$Index1(T, V1), Value1(V1), Index2(T, V2), Value2(V2), opposite(V1, V2) \rightarrow Phase(T)$

Here, $m1$ is more general than $m1$ and $m2$. The predicates that fit to instantiate the predicate variable $Index$ are the 13 indicators of the economy (see above), e.g., $lc(Time, Value)$ for unit labour cost. The predicates that fit to instantiate the predicate variable $Value$ are $low, medium, high$ and express the discretization of the real values of the indicators. The phase variable can be instantiated by $down, ltp, up, utp$ for four phases or by $down, up$ for two phases of the business cycle. The *opposite* predicates is used to write which qualitative value intervals are excluding each other. The background knowledge consists of such facts:

$opposite(low, medium).$
$opposite(high, medium).$
$opposite(high, low).$

Hence, the hypothesis space consists of all indicators or combinations of two indicators that allow to predict the phase of the business cycle. Uni-variate rules with just one indicator are excluded, because they are not considered to be sensible. Where $m2$ states that two indicators have to have values within the same range (e.g., both are low), $m2$ states that two indicators must have opposite value ranges. The three rule schemata here enforce the selection of the most informative indicators. By giving more complex rule schemata, the user enables RDT to learn more complex rules. The rule schemata are the means by which the set of interesting rules is specified.

RDT's learning procedure consists of two steps: hypothesis generation and testing. In a top-down, breadth-first manner, all possible instantiations of the rule schemata are generated and tested according to an acceptance criterion on the basis of all facts. For instance, the following rules which instantiate $m1$ and were learned in one of our experiments:

$mon1(T, V), medium(V) \rightarrow up(T)$
$lc(T, V), low(V) \rightarrow up(T)$

The following instantiation of $m2$ has been learned in another experiment:

$ic(T, V), medium(V), pc(T, V) \rightarrow down(T)$

A illustration for $m3$ is the following learned rule:

$rs(T, V1), medium(V1), x(T, V2)low(V2) \rightarrow down(T)$

If a rule has enough support but too many non supporting examples, it is considered too general. Later on, it becomes a partial instantiation of a more specific rule schema if this exists. If a rule does not have enough support, it is considered too specific. In this case, the rule need not be specialized further, since this cannot increase the number of supporting examples. RDT safely prunes the search in this case. RDT learns all valid rules that fit the rule schemata. Hence, a rule which is not learned, definitely does not hold given the facts.

2.1.2 Rule Inspection

For both, user given and learned rules, we might be interested in their coverage of examples and in their redundancy. The coverage can be measured by the percentage of the positive examples for a concept, that is covered by the rule. For instance, if there are 58 time points classified as *down*, we are interested in how many of them are covered by a certain rule predicting a downswing. We are also interested in the number of examples that are covered by all the rules together. The rule inspection of MOBAL indicates for each rule, how many of the input fatcs (in our experiments, time points that are classified into a certain phase of the business cycle) are correctly classified by the rule. For a set of rules, MOBAL indicates how many examples are covered by all the rules, or, in other words, how many examples cannot be explained by the rules. Redundancy of rule sets can be determined intensionally or extensionally. The intensional redundancy refers to the logical models that form the semantics of the rule set. The extensional redundancy of a rule set refers to the examples that are covered by the rules: if the same examples are covered by several rules, these rules are extensionally redundant [14]. Rules that are 100 % extensionally redundant are not necessarily intensionally redundant. They might cover new examples not yet seen which would not be explained by another rule. It is up to the domain expert to assess which of the extensionally redundant rules should be kept within the knowledge base.

2.1.3 Knowledge Revision

A set of rules can easily become contradictive. Most user-given rule sets first show contradictions because the user is not aware of all implications of all rules. Also learned rule sets can become contradictive. The most frequent contradiction occurs when applying the rules learned from a set of data (i.e. the training set) to another set of data (i.e. the test set): the predicted phase differs from the one given by the expert. In general, the detection and revision of inconsistencies is a hard problem. Due to the well-defined semantics of MOBAL and its restrictions of first-order logic, the problem could be solved [15, 16]. The system determines the facts and rules that are involved in the contradiction. It calculates all minimal changes to the knowledge base that would make it consistent again. The user chooses among the proposed changes and the system revises the knowledge base accordingly. Hence, the user is supported in building up a knowledge base about a domain by integrating rule sets, either learned or input. In particular, the user may input domain (causal) knowledge and the system watches that no learned rule contradicts the theoretical insight.

3 Experiments on German Business Cycle Data

Our leading question was whether ILP can support economists in developing models for dating phases of the business cycle. Given the quartely data for 13 indicators concerning the German business cycles from 1955 to 1994 where each quarter is classified as member of one of four phases, we used all but one cycle for learning rules and tested the rules on the left-out cycle. The leave-one-cycle-out test assesses the *accuracy* (how many of the predicted classifications of quarters corresponded to the given classification) and the *coverage* (how many of the quarters received a classification by the learned rules).

We now come back to the questions raised in the introduction. The learned rules automatically select pairs of relevant indicators. Hence, all learning experiments contribute to the question, which indicators actually influence the classification into one phase of the business cycle. Since the data we have are measuring the growth, the learning results refer to the growth cycle. However, we experimented with automatically finding ranges or levels of values of the indicators in order to base the learning results on a view of the business cycle which favours the level of indicator values (see Section 3.1). In order to tackle the question about the number of phases in business cycles, we have modeled four phases (see Sections 3.2 and 3.3) and two phases (see Section 3.4). An additional modeling decision needs to be made according to the handling of the time aspect in the data. In two experiments (Section 3.2 and Section 3.4), we just used the quarters as time points. No time intervals were formed. The rule schemata are the ones shown in Section 2.1.1. Hence, the rules only classify a quarter based on the measurements of this quarter. In a third experiment, we formed time intervals for the indicators and learned rules between them (Section 3.3).

3.1 Discretization

Before ILP can be applied, the originally real-valued time series of indicator values have to be transferred into discrete-valued temporal facts about this indicator. The goal of discretization is to provide the learning algorithm with data from which it can generalize maximally. This means, the discretization must be general enough such that rules learned from one situation can be transferred to another situation but specific enough such that non-trivial rules can be found. An example for a too specific discretization is to assign different values to every observation, an example for a too general discretization is to assign the same value to every observation. We use the number of generated facts to judge the quality of a discretization.

Actually, the task of discretization consists of two different subtasks:

Discretization of Values: split the continuous range of possible values into finitely many discrete values, e.g. by using equidistant thresholds or calculating suitable quantiles. For example, a gross national product of 4.93 in the fifth quarter could be expressed as the fact $y(5, 4.93)$.

Interval segmentation: for a given time series, find a segmentation of the time points into maximal sub-intervals, such that the values of the series in this interval share a common pattern, e.g. by approximating the time series by piecewise constant or piecewise linear functions. For example, the time series of gross national products $Y = (10.53, 10.10, 9.21, 5.17, 4.93)$ could be described as the temporal facts $y(1, 3, high), y(4, 5, medium)$, but can also be described as $y(1, 5, decreasing)$.

Interval segmentation can be viewed as discretization of the temporal values, therefore in this chapter we will use the name discretization as a generic term for both discretization of values and interval segmentation.

These two subtasks are closely intertwined: Discretized data can be very easily segmented by joining consecutive time points with identical discretization.

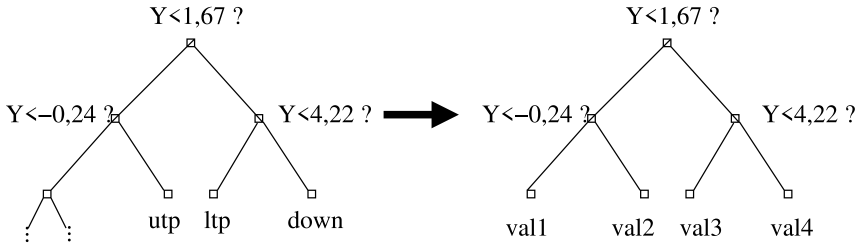

Figure 1: Decision tree and its induced discretization into $val1 \ldots val4$.

Also, segmented data can be discretized by building a discretization based on the patterns that lead to the segmentation. In this work, we chose the first approach to discretize the data, first because it is simpler and secondly because the indicators are already given free of trends (growth rates etc.), so it can assumed the relevant information lies in the value of the indicator.

To improve the quality of the discretization, we can also use the information that is given by the class of the examples [17]. In this case, we used C4.5 [11], a decision tree learner, to induce decision trees about the cycle phase based on only one indicator. The resulting trees were cut off at a given level and the decisions in this resulting tree were used as discretization thresholds. Decision trees of depth 2, i.e. using 4 discrete values, proved to build a suitable number of facts.

A closer look at the resulting discretization showed that in certain cases, the indicators had a very high variation, which lead to many intervalls that contained only one time point. In this case, the relevant observation may not be the value of the indicator, but the fact that this indicator was highly variating, i.e. no definite value can be assigned to it. This can be expressed by a new fact $indicator(T1, T2, unsteady)$, which replaces the facts $indicator(T1, T1 + 1, value_1), indicator(T1 + 1, T1 + 2, value_2), \ldots, indicator(T2 - 1, T2, value_n)$.

3.2 Modeling Four Phases Without Time Intervals

The data correspond to six complete businss cycles, made of four phases each. For the upper and lower turning point phases, no rule could be learned. Only for the upswing, each learning run delivered rules. Here are some examples of learned rules:

$gd(T, V), pc(T, V), low(V) \rightarrow up(T)$

stating that a low government deficit and a low consumer price index determine the phase as an upswing.

$c(T, V), l(T, V), medium(V) \rightarrow up(T)$

stating that a medium private consumption and a medium number of wage and salary earners classify a quarter as belonging to an upswing.

$rld(T, V), mon1(T, V), low(V) \rightarrow down(T)$

stating that a low long term interest rate and a low money supply can be used to date a downswing.

$rs(T,V), ic(T,V), medium(V) \rightarrow down(T)$
stating that a medium nominal short term interest rate together with a medium investment in construction point at being in the downswing.

Illustrating the rule inspection, we show the result for the first rule, called r171 in the leave-fifth-cycle-out learning run. The difference between the total number of facts about $up(t)$ and the input occurences of $up(t)$ is explained by the forward inferences of the (learned) rules. They derive further facts not given in the input.

```
****** Statistics on rule r171 ******

Number of rules with same conclusion -- up(t): 10

Coverage
  Total:
     Number of occurences of up(t): 77
     Number of occurences covered by r171: 43
     r171s coverage of all occurences: 55.8442 %
     Number of occurences covered by all rules: 59
     Total coverage of all occurences: 76.6234 %
  Of inputs:
     Number of input occurences of up(t): 59

     Number of inputs covered by r171: 35
     r171s coverage of inputs: 59.322 %

     Number of inputs covered by all rules: 41
     Total coverage of inputs: 69.4915 %
Redundancy
  Total:
     Number of occurences also covered by other rules: 28
     r171s internal redundancy (redundant/covered): 65.1163
  On inputs:
     Number of inputs also covered by other rules: 27
     r171s internal redundancy (redundant/covered) on inputs: 77.1429

************
```

If no rule states that the classification is exclusive, then no contradiction will be detected between the input fact $utp(31)$ and up or $utp(30)$ and $down(30)$. Hence, for testing, we entered rules of the form:

$utp(t) \rightarrow not(up(t))$

Then, we also find a misclassification.

```
Contradictory instances covered by rule r171:
auf(31) - [1000,1000]
```

In fact, time point 31 (corresponding to the second quarter of 1963) starts the upswing and time point 30 (first quarter of 1963) finalizes the downswing. The

Cycle	Accuracy	Coverage	No.of learned rules
LOO1	0.125	0.25	13 upswing
LOO2	0.5	1.0	12 upswing
LOO3	0.462	0.462	10 upswing, 2 downswing
LOO4	0.375	1.0	11 upswing
LOO5	0.696	0.696	10 uspwsing, 1 downswing
LOO6	1.0	0.36	1 upswing
Average	0.526	0.628	total: 60

Figure 2: Results in the four phase model using time points

first two quarters of 1963 are classified as the lower turning point. Misclassifications at the turning points are strikingly more frequent than in other phases.

For the downswing, only two learning runs, namely leaving out cycle 3 and leaving out cycle 5, delivered rules. Figure 3.2 shows the results.

The results miss even the baseline of 54% in the average. Leaving out the fifth cycle (from 1974 until 1982) delivers the best result where both, accuracy and coverage, happen to approach 70%. This might be due to its length (32 quarters), since also in the other experiment dealing with four phases the prediction of upper turning point and upswing is best, when leaving out the fifth cycle. Since the sixth cycle is even longer (45 quarters), we would expect best results in LOO6 which is true only for the accuracy this experiment. In the other experiment with four phases, the accuracy is best for upswing in LOO6 and second best for it in LOO5.

3.3 Modeling Four Phases With Time Intervals

Let us now see, whether time intervals can improve the learning results. We have used the discretization of the indicator values for the construction of time intervals. As long as the indicator value stays within the predefined level, the time interval is continued. As soon as the indicator value exhibits a level change, the current time interval is closed and the next one is started. We end up with facts of the form `Index(I,Range)`, and for each time point within the time interval I a fact stating that this time point T (i.e. quarter) lies in the time interval:I `covers(I, T)`. The chosen relations between time intervals were `iduring` and `overlaps`. The inverse of the regular *during* relation denotes a larger interval $I1$ in which somewhere the interval $I2$ starts and ends. `iduring(I1, I2)` is true for each time point within the larger interval $I1$. `overlaps(I1, I2)` is true for each time point of the interval $I1$ which starts before $I2$ is starting. More specific relations were not included in our model, because it is not very likely that the starting point of one interval is identical to the end point of another interval. The time intervals were calculation before the training started. The rule schemata were defined such that they link two indicators with there corresponding time intervals.

m1 (Index1, Value1, Value2, Phase):

$$Index1(I1, V1), Value1(V1), covers(I1, T),$$
$$Index2(I2, V2), Value2(V2), covers(I2, T) \rightarrow Phase(T)$$

Cycle	Accuracy	Coverage	No. learned rules
LOO1	0.166	0.166	73 upswing, 1 downswing, 2 ltp
LOO2	-	0	103 upswing, 3 downswing, 2 ltp
LOO3	0.375	0.352	87 upswing, 2 downswing, 2 ltp, 2 utp
LOO4	0	0	59 upswing, 7 downswing, 4 ltp
LOO5	0.355	0.344	88 upswing, 3 downswing, 4 ltp
LOO6	0.486	0.354	6 upswing, 2 downswing
Average	0.276	0.203	total: 450

Figure 3: Results in the four phase model using time intervals

m2 (Index1, Value1,Index2,Value2,Rel, Phase):
$Index1(T, V1), Value1(V1), covers(I1, T),$
$Index2(T, V2), Value2(V2), Rel(I2, I1) \rightarrow Phase(T)$

$m1$ substantially not differs from $m3$ in the preceeding model (time point model). It finds two indicators which determine the phase on the basis of a quarter which is shared by both time intervals.

$m2$ is more special in that it requires the time intervals of the two indicators to either overlap or include each other. Instantiations of $m2$ express rules where the behavior of one indicator must preceed or embrace the other indicator's behavior. These more specific rule schemata were intended to find rules for the turning phases, where no rules were learned in the previous experiment. In fact, rules for the upper turning point, upswing, and downswing were learned, but no rules could be learned for the upper turning point.

This rule states, that a period with high consumer price index growth, that is overlapped by a period of high growth rate in the private consumption, is indicatice of an upswing:
$pc(I1, V1), high(V1), covers(I1, T),$
$c(I2, V2), high(V2), overlaps(I2, I1) \rightarrow up(T)$
The next rule states, that a downswing happens, if during a period with medium growth in the number of wage and salary earners, the short term interest rate is high:
$l(I1, V1), medium(V1), covers(I1, T),$
$rs(I2, V2), high(V2), iduring(I2, I1) \rightarrow down(T)$

Another intention behind the time interval modeling was to increase the accuracy of the learned rules. Indeed, rules for the upper turning point could be learned with the average accuracy of 75% in the leave-one-cycle-out runs. However, the accuracy for upswing decreased to 34% in the average. Hence, overall the time interval model did not enhance the results of the time point model in as much as we expected (see Table 3.3).

3.4 Modeling Two Phases

In our third experiment we mapped all time points classified as upper turning point to upswing and all quarters of a year classified as lower turning point to downswing. We then applied the rule schemata of the first experiment. An example of the learned rules is:

Cycle	Accuracy	Coverage	No. learned rules
LOO1	0,8125	0,795	9 up, 69 down
LOO2	0,588	1,0	17 aup, 35 down
LOO3	0,823	0,571	2 up, 15 down
LOO4	0,8	0,35	6 up, 8 down
LOO5	0,869	0,8	10 up, 39 down
LOO6	1,0	0,701	6 up, 41 down
Average	0,815	0,703	total 50 up, 207 down

Figure 4: Results in the two phase model using time points

$ie(T, V1), low(V1), c(T, V2), high(V2) \rightarrow down(T)$
stating that a low investment into equipment together with high private consumption indicates a downswing.

Again, leaving out the fifth or the sixth cycle gives the best results in the leave-one-cycle-out test. Accuracy and coverage are quite well balanced (see Table 3.4).

These learning results are promising. They support the hypothesis that a two phase model is of advantage for the dating task. Concerning the selection of indicators, the learning results show that all indicators contribute to the dating of the phase. However, the short term interest rate does not occur in three of the rule sets. Consumption (both the real value and the index), net exports, money supply, government deficit, and long term interest rate are missing in at least one of the learned rule sets. For the last four cycles, i.e. leaving out cycle 1 or cycle 2, some indicators predict the upswing without further conditions: high or medium number of salary earners (l), high or mediuminvestment in equipment (ie), high or medium investment in construction (ic), medium consumption (c), and the real gross national product (y). It is interesting to note, that a medium or high real gross national product alone classifies data into the upswing phase only when leaving out cycle 1,2, or 4. Since RDT performs a complete search, we can conclude, that in the data of cycle 1 to cycle 4, the gross national product alone does not determine the upswing phase. Further indicators are necessary there, namely money supply ($mon1$) or comsumer price index (pc).

3.5 Concept shift

Starting from the two-phase model, we analysed the homogenity of the business cycle data. We want to know whether there are rules that are learned in all training sets, or, at least, whether there are rules that are more frequently learned than others. There is no rule which was learned in all training sets. Eight rules were learned from three training sets. There is one rule, which was learned in four training sets, namely leaving out cycle 1, cycle 4, cycle 5, or cycle 6:

$rld(T, V), l(T, V), low(V) \rightarrow down(T)$

We now turn around the question and ask: which training sets share rules? Eighteen rules were shared in the training sets leaving out cycle 5 and leaving out cycle 6. Four of the rules predict an upswing, fourteen rules predict a downswing. This means, that cycles 1 to 4 have the most rules in common. The data from the last quarter of 1958 until the third quarter of 1974 are more homogenious than all the data from 1958 until 1994. When leaving out cycle 1 or cycle 2, eleven rules occur in both learning results. This means, that cycles 3 to 6 have second most rules in common. The data from the second quarter of 1967 until the end of 1994 are more homogenious than all data together. This raises the question of the sample size for the dating problem [5]:page 16. Klinkenberg has investigated methods for handling *concept drift* by adaptively selecting the sample size for prediction and classification [7, 8]. Concept drift means that a concept changes over time. Concept shift is more specific and means that a concept changes at a certain point in time. Here, we investigate whether a concept shift has occured in business cycles.

We perform the same learning task on two disjoint data sets. We split the overall data set into two parts, cycles 1 to 3 and cycles 4 to 6. We apply training and leave-one-cycle-out testing to each part. The we check whether the increased accuracy is due to the smaller size or actually given by the homogenity of the data sets by putting together cycles from the two parts and see whether this also increases accuracy.

4 Conclusion and Further Work

Coming back to the questions asked in the introduction, our research has delivered some answers and some new questions. Let us start with the answers.

- ILP offers opportunities for the analysis of business cycle data. It is easy to interpret the results and the learned rules can be inspected with respect to redundancy and contradictions. The multi-variate nature of ILP and the automatic selection of most relevant indicators fits the needs of dating problem. However, numerical processes are not captured by ILP but a discretization must preceed ILP processing.

- Although some indicators are more dominant than others, no subset of the given indicators could be formed. All the thirteen indicators contribute to the dating of the phase.

- The two-phase model of the business cycle clearly outperformed the four-phase model. Where the best average accuracy in the four-phase model was 0,53%, the average accuracy of the two-phase model was 0,82%.

- There is a (???no) clear concept shift between cycle 3 and cycle 4 (around mid of 1971).

Indicators: what are the indices that form a phase of the cycle? more than two indicators in rule schema – not yet done

Which measurements of indices are to be taken? growth versus classical level cycle verschiedene Konjunkturmodelle hat [2] mit HMM untersucht.

discretization according to piecewise linear regression? – not yet tried, lack of interpretation

Background knowledge: Can ILP support economists in analysing cycles? causal knowledge (for knowledge revision etc. in MOBAL) needs be elicited from experts – not yet done

Influence of particular events can be inspected in ILP – not yet done

What is the appropriate number of phases in a cycle? Two phase partition: better partition into two phases according to the errors – not yet done

Concept drift vs. concept shift: concept drift or concept shift: drift cycle 3 and cycle 4 (1967,25 - 1974,50) or shift at end of 3, begin of 4 (1971,5) – further investigation needed (drift)

Acknowledgement This work has been partially sponsored by the Deutsche Forschungsgemeinschaft (DFG) collaborative research center 475 "Reduction of Complexity for Multivariate Data Structures". The authors thank Ullrich Heilemann, vice president of the Rheinisch-Westfälische Institut für Wirtschaftsforschung, for data of high quality and just as valuable suggestions as well as Claus Weihs and Ursula Sondhauss for raising our interest in the task and providing insight in its statistical nature.

References

[1] J. F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23:123–154, 1984.

[2] Marlene Amstad. *Konjunkturelle Wendepunkte: Datierung und Prognose.* St.Gallen, 2000.

[3] U. Heilemann and H.J. Münch. *West German Business Cycles 1963-1994: A Multivariate Discriminant Analysis.* CIRET-Conference in Singapore, CIRET-Studien 50, 1996.

[4] U. Heilemann and H.J. Münch. *Classification of German Business Cycles Using Monthly Data.* SFB-475 Technical Reports 8/2001. Universitaet Dortmund, 2001.

[5] Rheinisch-Westfälisches institut für Wirtschaftsforschung. *Arbeitsbericht 2000.* Rheinisch-Westfälisches institut für Wirtschaftsforschung, Essen, Germany, 2000.

[6] Jörg-Uwe Kietz and Stefan Wrobel. Controlling the complexity of learning in logic through syntactic and task-oriented models. Arbeitspapiere der GMD 503, GMD, mar 1991.

[7] Ralf Klinkenberg. Using labeled and unlabeled data to learn drifting concepts. In Miroslav Kubat and Katharina Morik, editors, *Workshop notes of the IJCAI-01 Workshop on* Learning from Temporal and Spatial Data, pages 16–24, Menlo Park, CA, USA, 2001. IJCAI, AAAI Press. Held in conjunction with the International Joint Conference on Artificial Intelligence (IJCAI).

[8] Ralf Klinkenberg and Thorsten Joachims. Detecting concept drift with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, pages 487–494, San Francisco, CA, USA, 2000. Morgan Kaufmann.

[9] Nada Lavrač and Sašo Džeroski. *Inductive Logic Programming — Techniques and Applications.* Number 148 in Artificial Intelligence. Ellis Horwood, Hertfortshire, 1994.

[10] K. Morik, S. Wrobel, J.-U. Kietz, and W. Emde. *Knowledge Acquisition and Machine Learning - Theory, Methods, and Applications.* Academic Press, London, 1993.

[11] John Ross Quinlan. *C4.5: Programs for Machine Learning.* Machine Learning. Morgan Kaufmann, San Mateo, CA, 1993.

[12] Anke Rieger. MP: An efficient method for calculating the minimum Herbrand model of chain Datalog programs. In W. Wahlster, editor, *Proc. of the Twelveth European Conference on Artificial Intelligence*, pages 385–389. John Wiley, 1996.

[13] Anke D. Rieger. *Program Optimization for Temporal Reasoning within a Logic Programming Framework.* PhD thesis, Universität Dortmund, Germany, Dortmund, FRG, 1998.

[14] Edgar Sommer. *Theory Restructering: A Perspective on Design & Maintenance of Knowledge Based Systems.* PhD thesis, University of Dortmund, 1996.

[15] Stefan Wrobel. *Concept Formation and Knowledge Revision.* Kluwer Academic Publishers, Dordrecht, 1994.

[16] Stefan Wrobel. Concept formation during interactive theory revision. *Machine Learning Journal*, 14(2), 1994.

[17] D.A. Zighed, S. Rabaseda, R. Rakotomalala, and Feschet F. Discretization methods in supervised learning. In *Encyclopedia of Computer Science and Technology*, volume 40, pages 35–50. Marcel Dekker Inc., 1999.