A Multiagent Approach for Logistics Performance Prediction Using Historical And Context Information

Yutao Guo, Jörg P. Müller
Intelligent Autonomous Systems
Siemens AG, Corporate Technology
Otto-Hahn-Ring 6
81730 Munich, Germany
joerg.p.mueller@siemens.com

Bernhard Bauer
Institute of Computer Science
University of Augsburg
Universitätsstraße 14
86135 Augsburg, Germany
bernhard.bauer@informatik.uni-augsburg.de

Abstract

This paper presents a multiagent architecture and methods for intelligent decision support in logistics processes. It extends current advanced prediction systems by providing the ability to combine history and situated reasoning. The contribution of the paper is threefold: first, a multiagent architecture and learning algorithms are developed that enables us to combine background models learned from history data with context-related knowledge about the current situation; second, using a large real data set we show that adding situated knowledge actually improves the performance of a supply chain decision support system; and third, for our settings we evaluate the degree to which agent-assisted decision support is actually usable/sufficient to improve human decision-making and to support automated decision-making in dynamic supply network management scenarios.

1. Introduction

In this paper we propose the notion of intelligent assistance for supply chain decision-making to tackle the complexity of cross-enterprise supply chain processes and to enable supply network collaboration. Our vision is that software agents act on behalf of organisations and humans, providing situated decision support and, ultimately, increasingly automating processes. To this end, agents need to perceive business environments, learn from their experience, and react to changes in these environments. At the same time, the use of peer-to-peer, multi-agent collaboration architectures supports flexible and self-organising supply network structures and decentral problem solving.

Over the past few years, there have been various research activities in this area (e.g., [8][6][3]). A number of business success stories have been reported where multinational companies applied agent technology to optimise their supply networks [10]. However, these are claims that can hardly be verified objectively, and the technical details of methods and implementations remain obscure.

State-of-the-art advanced planning systems [14] provide methods for process planning and prediction based broadly on history information. Our claim, triggered by observing real applications, is that this information, while being valuable, is not sufficient as a basis for intelligent supply chain decision-making assistance. Our claim is that history-based information should – and can – be enhanced by context-related situated knowledge to provide an improved basis for decision-making assistance.

Our objective is threefold: first, to develop a multiagent architecture and learning algorithm that enable us to combine background knowledge models (retrieved from analysing e.g., history data held in a business data warehouse) with context-related information about the current situation; second, to verify our claim that bringing in situated knowledge actually improves the performance of a supply chain decision support system; and third, to evaluate the degree to which agent-assisted decision support is actually usable/sufficient (a) to improve human decision-making and (b) to automated decision-making in dynamic supply network management scenarios.

The technical focus of this paper is on methods for intelligent agent assistance in logistics order fulfilment prediction. The agents in our system can sense the current context of transport order planning and execution, and they apply learned history knowledge models, e.g., identify critical orders with a high probability/impact of delay. This gives users sufficient time to develop contingency plans, and hence improves decisions. To combine history-based background knowledge with situated knowledge, we propose a multi-agent learning algorithm where history models are learned with Bayesian classifiers, and situated model is

integrated with the stacking strategy using Support Vector Machine (SVM) algorithm. The different models and functions are deployed as agent services in a multi-agent architecture. We evaluate the performance of our approach by using real order fulfilment data.

This paper is structured as follows: Section 2 summarises related research. Section 3 describes the multi-agent assistance architecture framework. In Section 4, the multi-agent learning algorithm is described. The learning model is evaluated with experiments on real data in Section 5. A discussion of the results, conclusion and perspectives of future research is provided in Section 6.

2. Related work

There're some initial research and applications of using software agents to provide assistance in supply chain management. TRACONET [11] is a vehicle routing system with an extended version of the contract net protocol. A multiagent approach for cooperative transportation scheduling is investigated in [3] to implement cooperation among the agents, task decomposition and task allocation, and decentralised planning. The joint research of SAP and Biosgroup focus on adaptive planning in supply chain management based on Ant Colony Optimisation (ACO) [12]. [4] describes an agent-based logistics monitoring architecture. On the commercial side, SAP offers an Event Manager component for their mySAP SCM product. The SAP Event Manager provides a basic distributed blackboard infrastructure together with a generic event model that enables parties in the supply network to publish events and to formulate responses based on perceived events. This solution can be regarded as a basis for the later deployment of intelligent agents. The research of this paper extends these approaches by proactive reasoning and prediction instead of mere passive monitoring.

The research in this paper on combining history and situated reasoning is related to *meta learning*. Meta learning learns from learned knowledge by assembling multiple learners. It is an active research area targeting improvement of learning performance and enabling distributed learning. Various meta-learning strategies have been proposed in the literature [2, 9]. *Voting* is the simplest method of combing predictions from multiple classifiers. In its simplest form, majority voting, each model contributes a single vote. The collective prediction is determined by the majority of the votes. In weighted voting, the classifiers have varying degrees of influence on the collective prediction that is relative to their prediction accuracy. Each classifier is associated with a specific weight determined by its performance on a validation set. The final prediction is decided by summing over all weighted votes and by choosing the class with the highest aggregate. The voting method combines the set of models in a linear fashion, which is not effective in some applications. The meta-learning method that we shall adopt and extend in this paper is *Stacking*, one of the most effective meta learning strategies [9]. It is described in Section 4.

3. Application scenario and system architecture

The application use case considered in this paper has been obtained from a major logistics service. This service is responsible for the administration of order delivery. Customers are business units who wish to deliver products to their customers, as shown in Figure 1. Orders are described by origins and destinations around the world, various product types from automation systems to medical equipment, and different packaging. To carry out an order, different local and international transportation service providers may be selected. Fulfilling an order often requires a combination of transport means and service providers. Due to high transaction volumes (several millions of orders over the period of one year), short planning/execution cycles, and a variety of unexpected events that happen during execution, monitoring each transaction manually is not feasible.

The mechanisms for agent assistance described in this paper are designed to effectively solve this problem by providing proactive monitoring and prediction throughout the order fulfilment process. In addition to keeping track of current events (e.g., a certain milestone has been reached with delay), the agents can learn from experience to provide users with advance warning that a problem is likely to occur. Considering the dynamic nature of the logistics process, we propose a learning approach that combines history based and situated reasoning, so that the agents can sense the current context and overlay this context information with previously acquired knowledge models.

The agents incrementally build two types of models: *history models*, based on past order execution records, and a *context model*, based on recently perceived data. When a new order is entered into the system, the combined models are used to provide users with information predicting future delivery status, including the likelihood of the order being delayed. The learning and model combination procedures are detailed in Section 4 and evaluated with real business data in 5.

Figure 1 illustrates the multiagent system architecture. Different agents represent different sources of events in the order fulfilment process (manufacturing, dispatching, delivery tracking) as well as the adaptive assistance functions (history model learners, meta learners). Based on interaction protocols, agents can collaborate with other agents, such as the service agents for customer notification, and the agents for tracking real time status of deliveries from the

transportation service providers' systems. This collaboration is beyond the scope of this paper that focuses on decision support aspects.

We found the multi-agent architecture an appropriate (if not the only applicable) approach to our problem as it provides us with the openness and flexibility required to add additional models, services, and collaboration patterns in the future.

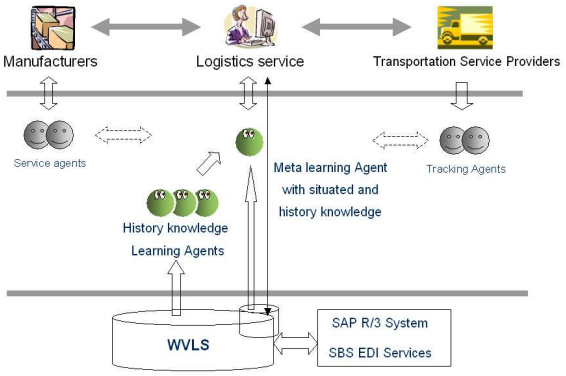

Figure 1. The system architecture of agent assistance in the logistics process.

Based on the described architecture, we have developed a protoype of a web-based agent-based decision-support tool for order dispatching and monitoring. Figure 2 illustrates the dispatchers' screen, showing a list of orders currently to be dispatched. This list is ranked by priority/urgency of orders, i.e., the dispatcher will find the orders that need to be serviced immediately on top of the list. By selecting an order, the order dispatching page will be displayed as shown in Figure 3. It supports a dispatcher in choosing an appropriate transport service provider. The agent on the right side of the screen provides suggestions for local and international transport service providers (labelled VSP and HSP) based on their history and current performance. The dispatcher will assign orders to transport service providers and commit the dispatching.

During order execution, events regarding the completion of milestones will be received, either via dedicated tracking agents or via a commercial system (e.g., SAP Supply Chain Event Manager). Based on this information and available history models, delay probabilities for individual orders are updated. In the prototype, we use the agent-based tracking system PAMAS [4] and its underlying milestone taxonomy. In this view, the set of orders visible to the user are sorted according to a mixture of delay probability and order priority. The user can select an order and inspect details on the order execution status, triggering actions such as notifying

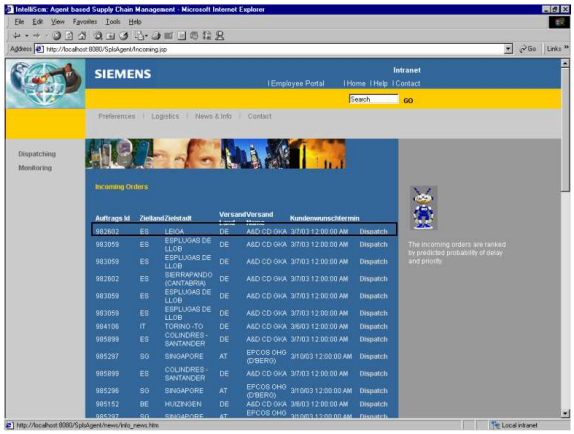

Figure 2. Screenshot of the order dispatching overview page

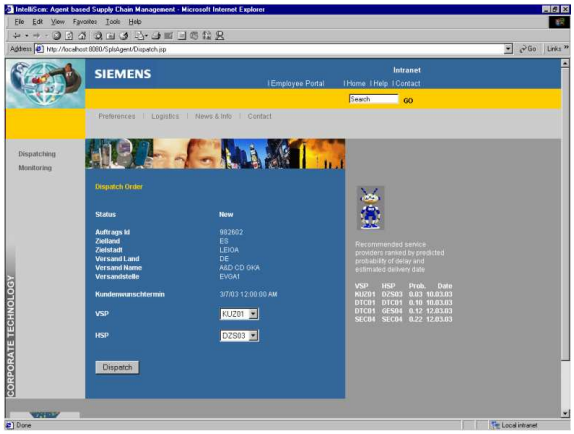

Figure 3. Screenshot of the order dispatching details page

the customer of delays, or re-assigning an order to a different transport service provider if possible.

4. Multiagent learning algorithm

4.1. Overview

The main idea of our research is to integrate history-based models with situated knowledge to improve prediction accuracy. Our technical approach to this integration task is based on the concept of meta learning (see Section 2 for an overview of related work on this topic). In the work described in this paper, we adopted an effective meta learning strategy called *Stacking* [17]. Stacking integrates independently computed base classifiers into a higher-level classifier by learning over a so-called *meta set*. A meta set

is generated by using the predictions of the base classifiers as attributes and keeping the original class labels. Then a $meta\ classifier$ can be trained with the meta set; this meta classifier is then used to classify unlabelled instances. The aim of this strategy is to correlate the predictions of the base classifiers by learning the relationship between these predictions and the true labels. So stacking allows effective non-linear combination of base classifiers. In this work, we added several improvements to the standard stacking method as described in [17]:

- Different algorithms are supported for history and situated knowledge learning: Bayesian classifier as the base classifier, and Support Vector Machine (SVM) as the meta learner. The reason for this is that situated learning needs to be done based on a much smaller training set than history-based learning. Support Vector Machines are an efficient learning method for learning problems based on small training sets.
- For the sake of improved learning performance, the meta set is generated with model predictions, as well as the original attributes of the orders.
- The meta set uses the predicted probability of delay instead of prediction labels (success, delay). This improves performance and eliminates the problem of determining the classification criteria.
- Only highly relevant attributes are selected to be the order attributes in the meta set, based on the results of a mutual information analysis (see Section 4.3). Not considering the less relevant attributes actually improves learning performance in our experiments.

The top-level approach of the learning algorithm is illustrated in Figure 4. It consists of the following steps:

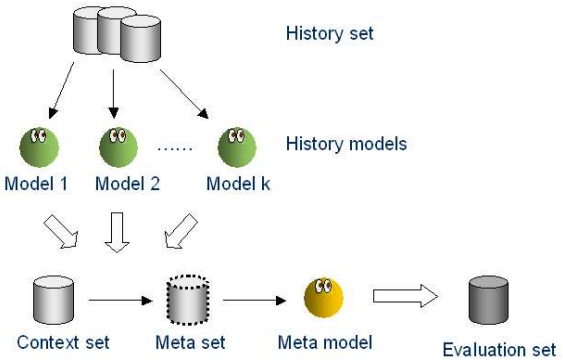

Figure 4. Overview of the learning approach

1. Learn k models $C_1,\ C_2,\ \ldots,\ C_k$ using the Bayesian classifier method from the history data set.
2. Select the $context\ set\ V$, containing the situated information to be considered. For the work described in this paper, we take a recent set of data (e.g., data from the current week) to represent the context set.
3. Generate a $meta\ set\ T$ with the context set and the predictions of the history models.
4. Learn a $meta\ model\ M$ on the $meta\ set\ T$ using the SVM algorithm.
5. Use the $meta\ model\ M$ to make predictions on the prediction set.

The individual learning steps are described in detail in the subsequent sub-sections.

4.2. History knowledge models

Most attributes of the orders have nominal-valued data types. Thus, using Bayesian classifier for learning from history data is appropriate. Bayesian classifiers assign the most likely class (e.g., delay, small delay, no delay) to a given instance (e.g., an order) described by its feature vector. The simplified Bayesian classifiers, Naive Bayes, assumes that attribute values are conditionally independent given the class values. Despite this assumption is unrealistic, Naive Bayes is remarkably successful in practice, often competing with much more sophisticated techniques [7]. In experiments, we were able to verify Langley's result by comparison with the classifier based on a Bayesian belief network. Therefore, we use the Naive Bayes method as the base classifiers for history knowledge:

$$\underset{y = y_j \in Y}{\operatorname{argmax}}\ P(y_j) \prod_{i=1}^{n} P(x_i|y_j) \qquad (1)$$

where y is the predicted class, $\mathbf{x} = (x_1, \ldots, x_n)$ is the attribute vector describing an instance to be classified, Y is the class value set. The various $P(y_j)$ and $P(x_i|y_j)$ terms can be estimated based on the training set. Given an instance with unknown label, the history model will output the probability of success/delay. Hence, a criterion is needed to provide explicit labelling. In this paper, we label the instances using a relative measure of delay instead of a fixed probability threshold. We achieve this by sorting the instances according to their probability of delay, and then labelling the top d percent of instances as *delay*. A realistic value for d is determined by computing the average percentage of delay instances in the training set. In the experiments reported later in this paper, the history models are built on a monthly basis.

4.3. Situated knowledge

As explained before, we represent the set of all available situated information by the so-called *context set*. The

context set and the learned history models are used to compute the meta set. In this paper, the attributes of a meta set data instance are composed of:

- the model predictions; and
- the original attributes of the orders.

Thus, we define the meta set T as $T = \{y, C_1(\mathbf{x}), C_2(\mathbf{x}), \ldots, C_k(\mathbf{x}), attribute\_vector(\mathbf{x}))|\mathbf{x} \in V\}$, where $\mathbf{x}$ is an instance in the $context\ set\ V$, y is the true class of $\mathbf{x}$, $C_i(\mathbf{x})$ is the prediction of model i on $\mathbf{x}$, and $attribute\_vector(\mathbf{x})$ contains the attributes of $\mathbf{x}$. The predicted probability of delay is used instead of predicted labels as the model prediction attributes, and only the highly relevant attributes are selected to be the order attributes in the meta set.

The computation of the relevance of order attributes is based on the information theoretical measure of *mutual information* [13].Mutual Information of two variables is defined as the reduction in uncertainty ($Entropy$) of the value of one variable given knowledge of the value taken by other variable. Mutual Information analysis has been successfully applied in feature-weighting in instance-based learning [1, 16]. The relevance of an attribute is determined by the mutual information $H(y; x_i)$ between the delivery status y and the value of attribute x_i of the order instance. Here, $H(y; x_i)$ measures the amount of information the attribute value conveys about the order delivery status. Continuous attribute values can be divided into predefined intervals [16]. The mutual information $H(y; x_i)$ between an instance class y and an attribute x_i is computed as:

$$H(y; x_i) = \sum_{v} \sum_{c} P(y = c, x_i = v) log_2 \frac{P(y = c, x_i = v)}{P(y = c)P(x_i = v)} \quad (2)$$

where $p(y = c)$ is the probability of delivery status being c, and $p(x_i = v)$ is the probability of value v for attribute x_i occurring in the respective training set. Table 1 shows the mutual information analysis results for the most relevant order attributes. It reveals that the destination attributes are most relevant to the delivery status, i.e., there is a relatively high correlation between the destination country of an order and its likelihood to be delayed.

4.4. Meta learning

In the last stage of our multiagent learning approach, a meta learner is trained with the meta set calculated in Section 4.3 to make delay predictions on an instance of an order. We use Support Vector Machine (SVM) classifier, which has strong mathematical foundations and excellent empirical performance in pattern recognition [15]. Based on statistical learning theory, the SVM learner has good generalization ability and reveals good performance even when the size of the available training data is limited. This enables us to integrate situated knowledge even if only a small context set is available.

Rank	Attribute name	Mutual Information
1	Destination Country	0.104
2	International Carrier	0.088
3	Origination Point	0.054
4	National Carrier	0.041
5	Container Type	0.033
6	Origination Country	0.007

Table 1. Top relevant order attributes by Mutual information analysis

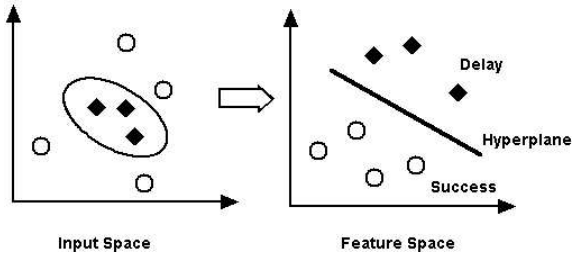

Figure 5. Meta learner using Support Vector Machine (SVM).

Figure 5 illustrates the principle idea underlying SVM: the SVM classifier projects data in the input space $\mathcal{X}$ into a higher dimensional feature space $\mathcal{H}$, with a kernel Function $k(\mathbf{t_i}, \mathbf{t_j}) = \phi(\mathbf{t_i})^T \phi(\mathbf{t_j})$, where $\mathbf{t_i}$ and $\mathbf{t_j}$ are instances in the meta set. Vapnik shows that the Structural Risk Minimization principle in statistical learning theory can be translated into finding the hyperplane with maximum margin for separating the positive and negative training instances. Computing this hyperplane is by solving the quadratic optimization problem in Equation 4 and 5. The solution will be a set of $support\ vectors$, which determine the position of the hyperplane. With the learned SVM model, the unlabelled orders can be classified as $success$ (positive) and $delay$ (negative), using the procedures illustrated in Figure 6. The decision function of SVM is

$$f(\mathbf{t}) = \text{sgn}(\sum_{i=1}^{m} y_i \alpha_i \langle \phi(\mathbf{t}), \phi(\mathbf{t_i}) \rangle + b) = \text{sgn}(\sum_{i=1}^{m} y_i \alpha_i k(\mathbf{t}, \mathbf{t_i}) + b) \quad (3)$$

$$\underset{\alpha \in R^m}{\text{maximize}} \; W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j k(\mathbf{t_i}, \mathbf{t_j}) \quad (4)$$

$$\text{subject to } \alpha_i \geq 0 \text{ for all } i = 1, \ldots, m, \text{ and } \sum_{i=1}^{m} \alpha_i y_i = 0 \quad (5)$$

where $\mathbf{t}$ is the order instance to be classified, $\mathbf{t_i}$ is the ith support vector and y_i is the true class of $\mathbf{t_i}$, m is the number of support vectors. Furthermore, by calculating distances of order vectors to the trained hyperplane, the SVM model can give a straightforward ranking of orders. In the experiments, as the meta set is unbalanced, i.e., the percentage of success orders is much higher than that of delay orders, so higher weight is given to the delay orders when training the SVM model.

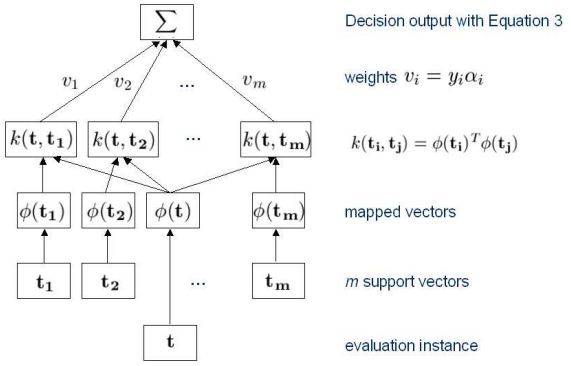

Figure 6. Making predictions with the SVM meta learner.

5. Empirical analysis

In this section, we describe experimental results to evaluate the performance of the proposed multiagent learning method with history and context information.

5.1. Experimental settings

The results reported in this paper were obtained by analysing large data sets from a major logistics service provider collected over a period of five months, from March to July 2003 each of which describes a single order execution including origin, destination, product type, packaging, local transport provider, international transport provider (where applicable), planned customer delivery date, planned pickup date, actual pickup date, and actual delivery date. Table 2 shows the size of the training sets used for every months as well as the size of context set and evaluation set.

Data set	number of orders
March 2003	365981
April 2003	390505
May 2003	404901
June 2003	422811
July 2003	406699
Context set	6088
Evaluation set	87994

Table 2. The data set size of the experiment

From this data, the data sets used by the meta learning approach described in the previous section were populated as follows:

- History models: five separate month models were learnt using the Bayesian classifier method, one for each March to July 2003.
- Evaluation set: approximately 88000 orders used for evaluation. These orders were August 2003 data except the first week in August, which was used as the context set.
- Context set: 6088 instances, randomly select 10% orders from the first week of August 2003.

Based on these models, we performed the following experiments on the evaluation set:

- E_1: A prediction algorithm that picks its decision (*Delay*, *Success*) randomly but compatible with the overall probability distribution of delayed orders. I.e., if the probability of an order being delayed is 15%, algorithm E_1 will answer it Delay in 15% of all cases. The purpose of E_1 is to know a lower quality bound produced by a prediction algorithm that only takes very little knowledge about the problem into account.
- E_2 to E_6: Predictions are made based on the Bayesian classifier history models from March to July, 2003, respectively.
- E_7: Predictions are made using one big single model containing the whole data set from March to July 2003 as the training set.
- E_8: Predictions are made based only on the situated model derived from the context set. I.e., history data will not be taken into account.
- E_9: Predictions are carried out based on the agent-based meta learning approach described in Section 4, combining the five month models used in E_2 to E_6 with the context model used in E_8.

Experiment	Learning model	Delay recall	Delay precision
E_9	Meta learning (history + context)	**0.66**	**0.48**
E_1	Distribution-based prediction	0.25	0.26
E_2	March'03 history model	0.41	0.38
E_3	April'03 history model	0.49	0.42
E_4	May'03 history model	0.43	0.38
E_5	June'03 history model	0.46	0.40
E_6	July'03 history model	0.43	0.39
E_7	Joint March to July'03 hist. model	0.50	0.44
E_8	Context set model only	0.40	0.37

Table 3. Summary of experimental results for meta learning and other models

5.2. Evaluation metrics and results overview

To evaluate the learning performance, we adopted two standard measures from information retrieval: $delay\ recall$ (DR) and $delay\ precision$ (DP), as defined in Equations 6 and 7.

$$DR = \frac{n_{D\rightarrow D}}{n_{D\rightarrow D} + n_{D\rightarrow S}} \tag{6}$$

$$DP = \frac{n_{D\rightarrow D}}{n_{D\rightarrow D} + n_{S\rightarrow D}} \tag{7}$$

where $n_{D\rightarrow D}$ is the number of correctly predicted *Delay* orders, $n_{D\rightarrow S}$ and $n_{S\rightarrow D}$ are the number of $Delay \rightarrow Success$ and $Success \rightarrow Delay$ errors. Intuitively, delay recall measures the share of actually delayed orders that the mechanism could successfully predict as *Delay*, whereas delay precision is a measure of the inverse noise ratio, i.e., the share of orders predicted by the system as *Delay* that were actually delayed. It is known from information retrieval that there is a trade-off between recall and precision, and that it is easy to optimise either recall or precision, but very hard to provide good results for both. Table 3 illustrates the experimental results.

6. Discussion of results, conclusions, and outlook

The first and obvious result of our experiments is that all learning algorithms that make use of history or situated knowledge perform better than the random prediction method E_1. That indicates that there are recurring patterns in supply network execution that lead to orders being delayed. Second, it shows that the learning performance of meta learning (E_9) based on history and situated knowledge widely outperforms all other methods. This reveals that single model learning is less effective in the dynamic logistics process and that even a relatively simple form of situated knowledge can lead to a considerable improvement. The fact that meta learning has better performance than the history models and the model trained with the context set shows that the meta learner applied the best knowledge of the history models by sensing the context set.

When examining the meta set, we made the interesting observation that different history models often come up with different opinions on the prediction of an instance. This is due to the dynamic and time-dependent nature of the logistics process and also indicates that meta-learning is beneficial in this application domain. The third result indicates that all of the algorithms considered(including the meta-learning) perform better in terms of recall than it does in terms of precision. While a 66% recall means that two third of all delays can be correctly predicted, the current precision level of less than 50% for the best algorithm indicates that there is a considerable number of *false friends*, i.e., orders that are mistakenly predicted to be delayed but which are not. Hence, while recall of the current meta-learning algorithm is appropriate for a decision-support algorithm, the precision ratio may not be acceptable due to a fairly high number of false alarms. An easy way of improving precision would be to increase the probability threshold for classifying an order as delayed. However, this would reduce recall.

In general the supply chain prediction method presented in this paper takes decision support approaches that are based on implicit and subsymbolic knowledge representation to its limits. We believe that the learning performance can be improved with the enhancement of explicit situated knowledge. These knowledge can be in an explicit factual or rule-based fashion (e.g., if a Thursday is a holiday, orders to be sent on the subsequent Friday in country X are likely to take a longer time to deliver), or that simply cannot be predicted (e.g., there is a railway strike in Austria from Wednesday to Friday, which will effect orders to, from, and across Austria). Integrating this explicit knowledge with the implicit knowledge approach described in this paper will greatly improve the agent prediction performance.

Another idea that we would like to explore in future work is to employ collaborative reasoning from different sources. For example, monitoring agents can enhance their reasoning with additional knowledge of the tracking agents used by the transport service providers. Multiagent learning in such a heterogenous and distributed business environment is investigated in another research [5]. Another extension of our system would be a more elaborate reasoning about root causes and context of a problem, and intelligent event response management, e.g., by presenting users appropriate recommendations for counter-actions.

Coming back to the three objectives that we set out with in this paper (see Section 1), we regard our results as first and important step towards intelligent supply chain assistance. We described an agent-based architecture framework

for combining history knowledge with situated information and we were actually able to show using real data that situated knowledge and the use of multiple models is actually beneficial in this domain. Even though our third hope – to achieve a prediction quality that fulfils the requirements of an online decision-support system – has not yet fully come true, we believe that a first step has been made and that we can improve our system sufficiently by adding a different type of model based on explicit factual or rule-based knowledge to our framework.

7. Acknowledgments

This paper is partly funded by the E.C. through the Athena IP. It does not represent the view of E.C., and authors are responsible for the paper's content.

References

[1] T. M. Cover and J. Tomas. *Elements of Information Theory*. John Wiley and Sons, New York, 1991.

[2] T. G. Dietterich. Machine-learning research: Four current directions. *The AI Magazine*, 18(4):97–136, 1998.

[3] K. Fischer, J. P. Müller, and M. Pischel. Cooperative transportation scheduling: an application domain for DAI. *Journal of Applied Artificial Intelligence*, 10(1), 1996.

[4] D. Frey, T. Stockheim, P.-O. Woelk, and R. Zimmermann. Integrated multi-agent-based supply chain management. In *Proceedings of the 12th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE-2003)*, pages 24–29. IEEE, 2003.

[5] Y. Guo and J. Müller. Multiagent collaborative learning for distributed business systems. In *Proceedings of The Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'04)*, New York, 2004. IEEE Press.

[6] M. N. Huhns, L. M. Stephens, and N. Ivezic. Automating supply chain management. In *Proc. 1st International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'02)*, pages 1017–1024. ACM Press, 2002.

[7] P. Langley, W. Iba, and K. Thompson. An analysis of bayesian classifiers. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 399–406. AAAI Press, 1992.

[8] T. Moyaux and B. Chaib-draa. Multi-agent coordination based on tokens: Reduction of the bullwhip effect. In *Proc.2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'03)*, pages 670–677. ACM Press, 2003.

[9] A. Prodromidis and P. Chan. Meta-learning in distributed data mining systems: Issues and approaches, 2000.

[10] N. Radjou, L. M. Orlov, and T. Nakashima. *Adaptive Agents Boost Supply Network Flexibility*, 2002. March 2002 Tech Strategy Brief.

[11] T. Sandholm. An implementation of the contract net protocol based on marginal-cost calculations. In *Proc. 11th National Conference on Artificial Intelligence (AAAI-93)*, pages 256–262, 1993.

[12] SAP AG. Managing the unmanageable with supply chain networks, 2001. www.sap.info/public/en/index.php4/article/comvArticle-193353c63af1170c55/en.

[13] C. E. Shannon. A mathematical theory of communication. *Bell Systems Technology Journal*, 27(1,3):379–423 and 623–656, July and October 1948.

[14] H. Stadtler and C. Kilger, editors. *Supply Chain Management and Advanced Planning*. Springer-Verlag, 2nd edition, 2002.

[15] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York, 1995.

[16] D. Wettschereck, D. W. Aha, and T. Mohri. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review*, 11:273–314, July and Oct. 1997.

[17] D. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.