Modeling the Column Recognition Problem in Tactical Information Fusion

Johan Björnfot and Per Svensson
Department of Data and Information Fusion
Division of Command and Control Warfare Technology
Defence Research Establishment
SE–172 90 Stockholm, Sweden
pers@sto.foa.se

Abstract - *We discuss the application of Hidden Markov Modeling (HMM) techniques to the column recognition problem, where a non-cooperative military unit consisting of a sequence of objects forms a transportation column. Here the task is to infer the object composition and organizational structure of the column from imperfect observations of individual objects, in combination with generic a priori information about the organizational structure of the non-cooperative forces. Good solution methods for the column problem would provide a significant contribution to the automatization of the force aggregation process in tactical situation assessment.*

Keywords: intelligence analysis, force aggregation problem, situation assessment, Hidden Markov Models.

1 Introduction

To make effective decisions, a tactical military commander needs the best possible information about both own and enemy[1] forces' situation and options. Historically, the decision maker has had limited means at his disposal to help him assess and understand his tactical situation on the basis of all available and relevant factors. Today, however, a large number of information collection and processing devices are available to aid the commander.

Although on one hand the current rapid technological evolution of sensor, communication, and information processing systems has led to an ever larger availability of intelligence information, on the other it creates a need for a faster tempo of operations and therefore of tactical decisions and situation assessments. In international peace-enforcement operations additional complicating factors occur: need for political acceptance by a possibly sceptical international public opinion, partially opposed local civilian populations, low density of own forces relative to those of the opponent, high confidence levels required for target engagement. To help tactical intelligence analysts cope with these requirements, more efficient and less labor intensive computer-aided situation and threat assessment methods are needed, helping them to timely combine uncertain information fragments into coherent and increasingly complete knowledge representations.

After introducing in Section 2 the concept of *tactical information fusion*, we survey previous work concerning this concept with some focus on one of its subtasks, the *aggregation process*. This is the process of fusing information about the composition of entities on a low organizational level with generic apriori knowledge of organizational structure to infer the higher-level composition of non-cooperative forces, a key task in ground combat *situation assessment*.

In Section 3, we describe the main contribution of this paper, i.e., the application of *Hidden Markov Modeling* (HMM) techniques [1],[2],[3] to the *column recognition problem*, a special case aggregation process applicable where a non-cooperative military unit forms a transportation column. Again, the task is to infer the object composition and organizational structure of the column from imperfect observations of individual objects in combination with generic a priori information about the organizational structure of the non-cooperative forces.

The results stated in this paper are elaborated further in [4], the M Sc thesis of one of the authors.

2 Tactical information fusion

Data fusion is a broad concept, containing the processes *object refinement* (level 1), *situation assessment* (level 2), *impact assessment* (level 3) and *process refinement* (level 4) [5]. In this paper, we use the term *information fusion* to denote the later stages of the data fusion process, i.e. levels 2-4.

The purpose of object refinement is to achieve greater robustness, precision and range by combining information, possibly from several sensors of various kind.

In situation assessment one tries to identify the observed situation by inferencing from observed data and events in combination with relevant a priori information. Several alternative hypotheses are generated and evaluated.

In impact assessment risks and opportunities in each

1. Henceforth, we will preferentially use the more general and less antagonistic term *non-cooperative* forces

of the inferred possible situations are analysed and evaluated.

Process refinement is a feedback process aiming at improving the result of a data fusion process by controlling associated collection and interpretation processes so as to reduce identified (and significant) uncertainties in the fusion product.

By *tactical information fusion* we denote here the use of information fusion processes in military tactical scenarios, in particular near-real-time interpretation of intelligence information relating to large-to-medium size formations of ground forces (division level and below).

There are few detailed accounts of information fusion systems and methods in the open literature, partly due to confidentiality restrictions, partly to the fact that the number of such methods and systems is still limited. Below, we survey three recent papers which summarize well the state-of-the-art open literature on tactical information fusion.

2.1 NATO Data Fusion Demonstrator

A group of NATO countries have developed the *NATO Data Fusion Demonstrator (DFD)* [6]. The DFD consists of a scenario simulator, a fusion system (DFS), and a result evaluator. The simulated scenario generates a flow of intelligence reports which is sent to the fusion system. A set of fusion methods can be applied to this flow in combination with various kinds of apriori terrain and doctrine information to produce a situation picture which is then compared to the ground truth situation by the result evaluator.

The fusion process in DFS consists of a set of subprocesses:

• *classification* is a process in which the kind and size of an observed unit is determined from knowledge of the opponent's doctrine

• *correlation* is a process which combines information from different sensors which refers to the same object; this process is used in particular to track moving objects

• *aggregation* is a process which combines related objects using apriori knowledge of the opponent's organization structure ("order of battle")

• *fusion* is a process which combines incomplete and uncertain information from different sources relating to one or more objects

• *situation assessment* is a process which produces the most complete and consistent hypothesis for the tactical situation picture based on all the information from the previous processes.

Evaluations of the DFD indicate that its technology provides several opportunities to increase the degree of automation of the tactical intelligence process. On the other hand, further research and development is needed to create an operational system based on the DFD technology. In particular, the aggregation problem is indicated as an issue requiring further research.

2.2 Hybrid AI architecture for information fusion

The paper [7] describes a system prototype based on combining fuzzy logic and bayesian networks. The system aims at fusing available information, generating hypotheses about what situation may have caused the intelligence information, predicting probable future actions and analyzing needs for complementary sensor information.

O*bject identification* is a continuous process controlled by a *fuzzy logic module* where incoming data are associated with previously observed objects. New estimates are made of the objects' type and position. The updated object information is stored in the database.

The *situation assessment process* has two purposes:

• to identify probable situations which may have caused the observed information

• to generate hypotheses about possible future events.

Situation identification is obtained by letting a *fuzzy logic event detector* compare the observed behavior of objects with predefined events stored in an event library. Objects which probably belong to the same unit are associated. Labeled by their estimated probabilities, the situations identified are then passed on to a bayesian network. Its purpose is to maintain and evaluate a small set of predefined hypotheses about possible enemy intentions. The probabilities associated with each hypothesis are updated whenever new or corrected information arrives from earlier stages of the fusion process.

Feedback is obtained by having a *fuzzy logic collection module* suggest how available resources are to be used to obtain the desired information. These proposals are based on a knowledge base of more than 100 rules and several fuzzy variables, in combination with information about characteristics and positions of available sensors and sensor platforms.

Testing of the prototype was done by simulation. Results of the simulation are described as satisfactory, e.g., the system was able to aggregate lower level units into higher level forces. No quantitative corroboration of this claim is given in the paper.

2.3 Bayesian network-based recursive composition inference for force aggregation

The paper [8] proposes a solution for the aggregation problem. The authors use a bayesian network to make inferences of probable unit types from observations of individual vehicles. The method employs models representing types of vehicles used by different units, in combination with estimates of the probability $P(y|x)$ to observe a vehicle of type y given that its actual type is x. It is assumed that all vehicles observed belong to the same unit. The paper briefly discusses the possibility to extend the model to cover also the combination of propositions about several smaller units into hypotheses about the composition of more complex force units. The only parameter studied in the model is vehicle type. No reference is made to vehicle formations, such as columns.

In [4] we briefly studied the use of classical bayesian-network based methods for the force aggregation problem. We found that these methods should be useful primarily to aggregate unordered clusters when primary object (vehicle) classification can be considered to be accurate (non-probabilistic) and only the inferences regarding higher level units are uncertain. We believe that application of bayesian network methodology to force aggregation should be based on learning from data [9], since the problem of determining all required conditional probabilities will probably otherwise become overwhelmingly difficult. By use of learning techniques, the above-mentioned requirement of accurate primary object classification could perhaps also be relaxed.

3 The aggregation process and the column recognition problem

In general, the aggregation process does not presuppose any particular context in which the non-cooperative forces occur. The only assumption necessary is that organizationally close vehicles tend to be strongly clustered in some observable space, geographical, temporal or other.

There may exist situations where the likelihood for the existence of a specific unit type could be high even if only a small part, possibly even a single piece of equipment, has been observed. Although one should not expect such situations to occur often, for example the spotting of a specific kind of missile launch vehicle might be highly significant if and when it occurs. So one might ask how large a subset of the components of a certain unit type must have been observed in order to justify an aggregation attempt. It turns out that this issue can be left to the modeling method itself to decide.

In situations where clustering is associated with a preferred ordering of objects, much stronger conclusions about the character and composition of the entire structure can be drawn from a given number of observed objects, given that the structure approximately conforms to one of a set of alternatives known a priori. The prototypical problem here is recognition of kinship between stretches of DNA from different organisms [1],[2]. The importance of this problem has given rise to a new branch of science, *genome informatics*, a subarea of bioinformatics.

This paper takes inspiration from results in genome informatics to find ways of recognizing (parts of) an organization from imperfect observations of sequences of objects. We call this the *column recognition problem*, in the expectation that our model fits the intelligence problem of recognizing ground troop organizations from (sub)sequences of observed object types.

We restrict ourselves to situations where we have available a set of observations of objects in a ground target environment. A comparatively simple and unambiguous scenario is that of a complex military unit moving as a column along a road or road network. The observations may originate from various sources, such as human observations, radar sensors, signal intelligence etc. The problem is to generate a hypothesis distribution of possible force compositions from these observations in conjunction with a priori information about the appearance, transportation behaviour, and organization of the non-cooperative forces.

3.1 A priori knowledge in the form of military doctrine

The behaviour and formation of a military unit or force is usually controlled by a set of more or less strict rules, a *tactical doctrine*. These rules define, e.g., the composition of units, as well as distances and ordering between neighbouring units. The set of rules usually depends strongly on which tactical operation the military unit is performing.

The doctrine describes the typical preferred behaviour of each kind of unit. Actual situations may show smaller or larger discrepancies since the behaviour of a unit is influenced not only by doctrine but also by situation-dependent personal preferences of the force commanders. In addition, various attempts to achieve deception may occur. E.g., the unit may try to hide its identity or indicate a different tactical objective than the real one. Weather and geography are additional factors which may influence the behaviour of the unit.

Below, a rough classification of different rules of formation is made depending on how strictly the structure of the column is controlled by doctrine. Presumably, any real case will fall somewhere “in between” these classes.

- **Complete freedom with regard to ordering:** The order between vehicles is random, i.e. any sequence of input symbols to the interpretation process is equally probable.
- **Clustered unordered units at each level:** In this case all components of individual subunits cluster together but no predetermined order between units on the same level is enforced.

Figure 1: **Controlled order for lower-level entities, unordered on higher levels:** lower level units (e.g., platoons) tend to maintain a predetermined order between

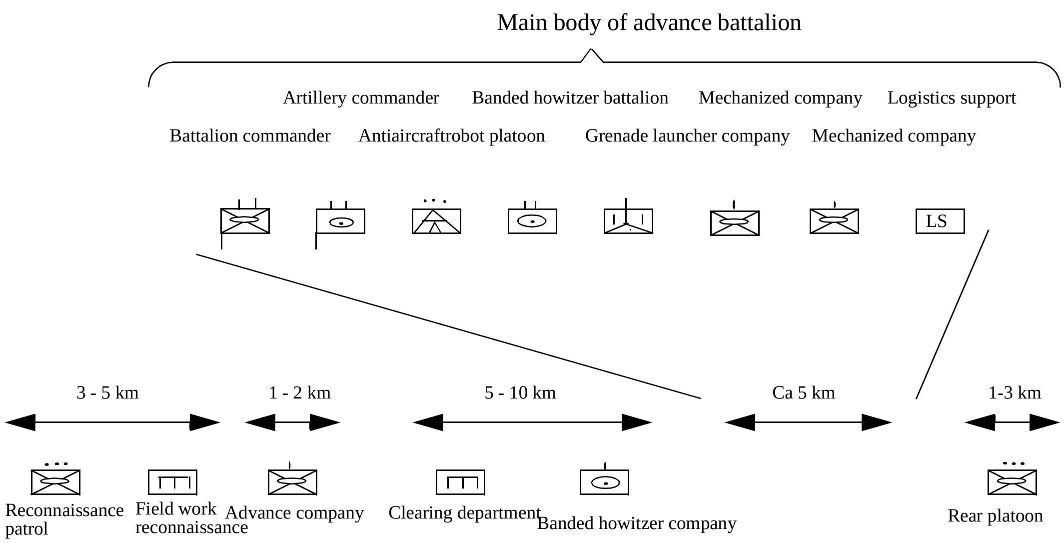

Figure 1: Deployment of an advance battalion, Russia.

its components (e.g., vehicle types), whereas higher level units, (e.g., companies) do not.

• **Controlled order for higher-level entities, unordered on lower levels:** higher level units tend to maintain a predetermined order between its components, lower level units do not.

• **Completely controlled:** the formation on any level of hierarchy is determined by doctrine. Thus the sequence of symbols that make up the sensor classifications is deterministic.

• **Probabilististically controlled:** for each subunit of the column there exists probability distributions describing possible formations and compositions.

Different combinations of these cases might also be considered, e.g., where some units on the same level are ordered, others unordered.

3.2 The observation process

There exist many different types of information collection system that may be used to obtain information about, or detect, non-cooperative force structures. Tactical intelligence collection is often obtained through one or more of the following general techniques:

• Signal reconnaissance
• Radar reconnaissance
• Air reconnaissance
• Troop reconnaissance, e.g. using special forces.

These techniques are neither exhaustive nor mutually exclusive and they should only be considered as practically important examples.

Here we concentrate on the situation analysis part of the data fusion process. We usually assume that data from the collection process have been subjected to the *object refinement* process. Thus, a first processing of the input has been performed where uncertain information from one or several sensors has been combined into a refined sequence report about the observed objects. Sensor performance aspects such as sensitivity, range etc. have been taken into account at this stage.

The fact that observations are uncertain means that some of the sensor propositions about type, position, or time are uncertain. The resulting information for each potential vehicle in the observed sequence is represented as a probability distribution over possible vehicle types. If unusually large distances between consecutive vehicles need to be considered during interpretation, *delimiter objects* could be interspersed and formally treated as a special kind of vehicle.

Due to limited sensor coverage, geographical obstacles and other factors which may influence observations it is plausible that only a subset of the force has been observed.

To aggregate these uncertain object observations means to cluster objects which are believed to belong to the same unit. In the ground force scenarios we consider here, this usually means clustering first object observations into platoons, then platoons into the next hierarchical organizational level, companies, etc. The aggregation principle is either always the same, or it may exploit different a priori knowledge about ordering at different levels, as discussed above.

4 Modeling the column problem with Hidden Markov Models

The method we employ here is *Hidden Markov Models (HMM)*, see [1],[2],[3]. Sequence recognition using HMM is based on the following general idea: given a set of sequences of symbols, or *strings*, with possibly imprecise similarities in structure. Such a set is said to form a *family*. From a given family of sequences it is possible to create an associated HMM by “training” the model using data from the family. The HMM may then be used e.g. to decide if an arbitrary string belongs to the family.

A typical application is matching of DNA strings in genome informatics [1],[2].

Application of HMM to the column problem requires the availability of a priori “doctrinal” information describing the normal formation of units with respect to object type and mutual ordering. The method thus starts by defining the structure of a model by which the doctrinal knowledge is to be represented. The model is then trained using a set of data, either generated from doctrines using simulation and/or obtained from real observations.

An observed sequence may then be assessed against a bank of trained models, each member of which obtained from training data (approximately) corresponding to a specific doctrine. The assessment is made by computing a *measure of fit* for the sequence, based on the probability of observing the sequence given the model, see below and [1],[2],[3].

If observations are given as a probability distribution over possible object types one may use Monte Carlo simulation to generate a set of sequences from these distributions. Each one of the sequences are then assessed against the model. The mean value of the measure of fit from the different simulated sequences may be used to estimate the likelihood that the observations originate from the type of unit which the model was trained to recognize.

4.1 Structure and properties of Hidden Markov Models

An HMM is a stochastic process composed by two related probabilistic mechanisms. These are (1) an underlying Markov chain with a finite number of states and (2) a stochastic symbol generating process associated with each state. For each discrete time step the process is assumed to be in some state, and a symbol is generated by the associated stochastic process. Then the Markov chain changes its state according to its associated transition probability matrix. Only the generated symbols but not the state of the Markov chain can be observed, thus the name *Hidden Markov Model*.

A so-called Standard HMM Architecture is shown in Figure 2.

The model can be viewed as a non-deterministic finite state machine which generates strings from an alphabet $O = \{o_1, o_2, \ldots, o_K\}$. The nodes in the HMM correspond to different states. In addition to the *start* and *end* states there are three kinds of state: m_i are *main* states, d_i are *delete* states, and i_j are *insert* states. The main and insert states always emit a symbol, in our basic example one of the vehicle types in the column. In the delete state no symbol is emitted. Edges correspond to transitions between states. Each node (state) has an associated probability distribution for the different transitions that may occur in this state. The main and insert states also have probability distributions for emitted symbols. These distributions are represented as transition and emission probability matrices of the model. The *length* of the model is defined as the number of main states. The HMM of Figure 2 has length 2.

More formally, an HMM is characterized by:

• **Underlying Markov chain**: A Markov chain $\{X_n\}_{n=0}^{\infty}$ taking values from a finite set of states $S = \{1, 2, ..., J\}$ with J states. The transition probabilities between states are given by the transition matrix $A = (a_{i|j})_{i=1, j=1}^{J, J}$, where $a_{i|j}$ is the probability for transition from state i to state j, subject to:

$$a_{i|j} \geq 0, \sum_{j=1}^{J} a_{i|j} = 1,$$

At time $n = 0$ the initial state X_0 is given by the probability distribution $\pi(0) = (\pi_1(0), \ldots, \pi_J(0))$ where $\pi_j(0) = P(X_0 = j)$.

An HMM is often modelled as having specific *start* and *end* states. Since in this case the Markov chain always starts in the *start* state, we get $\pi_{start}(0)=1$. The *end* state becomes an *absorbing* state with $a_{end|end}=1$.

• **Observable stochastic process**: A stochastic process $\{Y_n\}_{n=0}^{\infty}$ with a finite set of states $O = \{o_1, o_2, ..., o_K\}$, where normally $K \neq J$. The processes $\{X_n\}_{n=0}^{\infty}$ and $\{Y_n\}_{n=0}^{\infty}$ are for each fixed n related by the conditional probability distribution $b_j(k) = P(Y_n = o_k \mid X_n = j)$. These form the *emission probability matrix* B according to:

$$B = (b_j(k))_{j=1, k=1}^{J, K}$$

subject to $b_j(k) \geq 0, \sum_{k=1}^{K} b_j(k) = 1$

• **Conditional independence:** For each sequence of underlying states $j_0 j_1 \ldots j_n$ the probability of observing the string $o_0 o_1 \ldots o_n$ is given by:

$$P(Y_0 = o_0, \ldots, Y_n = o_n | X_0 = j_0, \ldots, X_n = j_n, B) = \prod_{l=0}^{n} b_{j_l}(l)$$

The probability distributions for observing some string o_i is given by the parameters A, B, and $\pi(0)$, compactly represented as $\lambda = (A, B, \pi(0))$ [3].

Initial transition and emission distributions are obtained from some kind of a priori knowledge. One might e.g. define the initial probability for the emission of a “lorry” as being equal to the a priori probability for lorries, i.e. if we know that an army has N vehicles of which n are lorries this probability is n/N, corresponding to a so-called *Dirichlet distribution* [1]. If there is no a priori knowledge the initial probabilities are equally distributed over reachable states.

The network is trained by being fed a family of strings corresponding to the doctrinal knowledge, using the *Baum-Welch* algorithm [3]. During training the probability distributions are modified to fit the family of strings generated from the doctrinal knowledge.

4.2 Sequence assessment

To assess how well an observed string fits a given family the measure of fit for the string is computed using the HMM. Given that the HMM has parameters $\lambda=(A,B)$, where A and B are the transition and emission probability matrices respectively, having been created from a family of strings, how does one decide if an arbitrary string $o_0 o_1 \ldots o_N$ belongs to the family?

This can be done by computing the probability $P(Y_0 = o_0, \ldots, Y_N = o_N \mid \lambda)$ for the string, i.e. the probability to observe the string given the model. This probability is called the *measure of fit of the string given the model*.

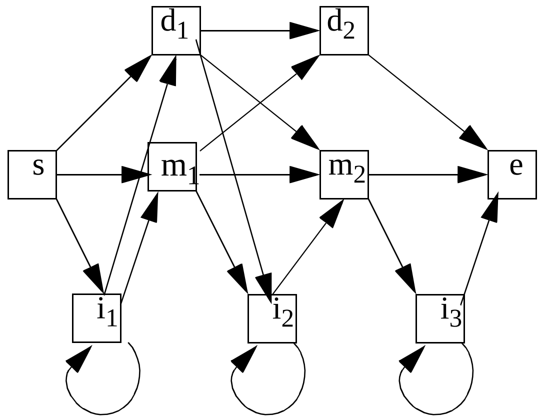

Figure 2: HMM Standard Architecture.

How is $P(Y_0 = o_0, \ldots, Y_N = o_N \mid \lambda)$ computed? To do this we first define the *forward variable* $\alpha_n(j)$ giving the probability of observing the string $o_0 o_1 \ldots o_n$ up to time step $n \leq N$ and that the hidden Markov model at time step n is in state j, $1 \leq j \leq J$:

$\alpha_n(j) = P(Y_0 = o_0, \ldots, Y_n = o_n, X_n = j)$.

It can be shown, see [3], that $P(Y_0 = o_0, \ldots, Y_N = o_N \mid \lambda)$ may now be computed as:

$P(Y_0 = o_0, \ldots, Y_N = o_N \mid \lambda) = \alpha_N(j)$

When comparing a string, o, against an HMM, M_i, one obtains an estimate for the probability that the string was generated by the model, $P(o|M_i)$. Thus by comparing an observed sequence against a bank of HMMs corresponding to several possible strings, a sequence of probabilities $P(o|M_1)$, $P(o|M_2)$, $P(o|M_3)$, ... is obtained, expressing the likelihood that each model has generated the sequence.

From this we want to obtain a probability distribution for the different units given the observation o, i.e. $P(M_1| o)$, $P(M_2| o)$, $P(M_3| o)$,..., subject to $\sum_i P(M_i|o) = 1$.

This can be computed from Bayes rule as:

$$P(M|o) = \frac{P(o|M)}{P(o)} \cdot P(M)$$ where $P(o)$ is the probability

for making the observation. The exact value of $P(o)$ is inconsequential since this factor is present in all terms when performing the normalization as described below. We therefore define: $Q(M|o) = P(o|M) \cdot P(M)$

$P(o|M)$ are the probabilities obtained from comparisons with the HMM. $P(M)$ are the a priori probabilities for the string which corresponds to the model M. Without any prior information on the origin of the observations these probabilities are set uniformly for all possible strings.

In order for the computed values to form a probability distribution they need to be normalized as:

$$P(M_j|o) = \frac{Q(M_j|o)}{\sum_i Q(M_i|o)}$$

The computed measures of fit represent the probabilities given the observation for the HMMs corresponding to each string.

These measures of fit give, however, only a measure for how well the entire string, or observation sequence, corresponds to each unit. A measure of how well different partial sequences fit the model is also desirable. In [4] we suggest the use of the *Viterbi* algorithm [3] to support this purpose. This algorithm detects the most probable path through the HMM and may thus be used to estimate the number of main states traversed while matching different strings.

4.3 Alternative ways of applying HMM to the aggregation problem

Assume that a unit has been partially observed as in Figure 3. For simplicity we assume that one has observed four out of seven objects making up the unit. We now want to compare the observations 1´- 4´ against a set of HMMs corresponding to different units which the column 1-7 may represent.

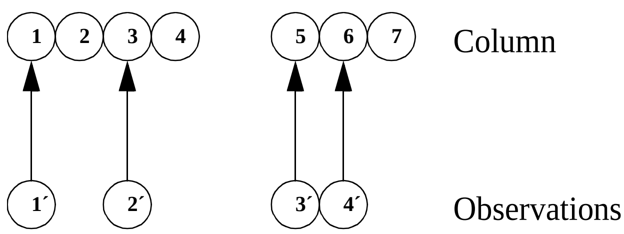

Figure 3: Observations of a column.

4.3.1 Aggregation in a single step

In this case one tries to estimate the total composition of the column directly from the observations. Assuming that we have observed parts of a battalion this means that we try to estimate the likelihood of different types of battalion directly from the observations without estimating their constituent platoons and companies first.

We summarize the major subprocesses when aggregating in a single step as follows:

• Generate training sequences from doctrine for the unit

• Create HMMs for different unit types through training using these sequences

• Select one or more observation sequences from the set of observations

• Compare the observation sequences with the different HMMs to decide which unit type the observations fit best.

4.3.2 Aggregation for each level of hierarchy

In this case aggregation is made in several steps, one for each level of the hierarchy. The results are then used to estimate the complete force structure. This means that an HMM has to be created for each subunit. For this the same conditions apply as when aggregating in a single step. To perform the aggregation one also has to decide which objects may be assumed to belong to the same unit.

We summarize the major subprocesses as follows:

• Generate training sequences for the unit from doctrine

• Create HMMs for different unit types on each level of hierarchy through training from these training sequences

• Select one or more observation sequences from the set of observations

For each aggregation level of hierarchy the following is done:

• Cluster objects into possible formations and compare how well each cluster corresponds to HMMs for the level one wants to aggregate to

• Create sequences for the new level using the result from the previous step. Go to the previous step if not ready.

5 Test results

Since we have not had access to empirical data on the formation of units, we have instead made a number of simulation tests using Monte Carlo-generated input strings.

Simulations were designed to correspond mainly to those of the cases described in 3.1 which should be best suited to modeling using HMM [4].

Assume for simplicity that company subunits (platoons) are elementary detectable objects. Each subunit is represented by a single lower-case letter in the input string. Let us assume that *T=tts* corresponds to an armoured company, *A=aaas* to an anti-aircraft company, *M=mms* to a mechanized company, and *L=llsl* to a logistics support company, where *t* is a tank platoon, *s* a staff platoon, *a* an anti-aircraft platoon, *m* a mechanized platoon, and *l* a logistics support platoon.

The case where companies are clustered and the order both within and between companies is random was simulated as follows. The components of each "company" string were randomly permuted, then the resulting strings were randomly joined. Training was made using 5000 such strings, which is the approximate number of different strings in this example. It turns out that the resulting measure of fit is unable to discriminate between different strings. The method is thus unsuitable in this case, which should come as no surprise since only clustering and no ordering is involved.

The case where larger units are unordered and smaller units ordered was tested in a similar way, requiring only 200 strings in this case. The measure of fit was about 10^{-6}, essentially independent of the order between companies, i.e., higher level units.

The "opposite" case where larger units are ordered, smaller unordered, gave a higher measure of fit than the previous case, about 10^{-3}.

Although the values for the measures of fit may seem small, the method should be useful in both these cases if properly trained since it is able to discriminate between strings that follow the training pattern and those that do not (giving them much lower scores).

Sequences derived from strings faithful to doctrine by insertions or deletions were also found to be recognizable, although the creation of an appropriate collection of training sequences requires careful experimentation and analysis.

Finally, a two-level hierarchical HMM was tested for the case of an ordered "battalion" with unordered "companies", allowing both insertions and deletions. The results obtained in this case were encouraging, with a measure of fit about 0.1 and responding to insertions and deletions in a way consistent with intuition.

In all cases, the length of the trained model was obtained as the average length of the strings used to train the model.

6 Discussion and conclusions

The general conclusion we draw from our tests is that the method seems applicable to the problem at hand, given that some information about the order between objects is available a priori. The applicability of the method increases with the degree of order of the units. An interesting result is that cases with a high degree of order on higher hierarchical levels and lack of order on lower levels give better measures of fit and are therefore better suited to the method than vice versa. Our test results indicate that the best method is to make aggregation separately for each level of hierarchy. This reduces computation time and model fit is improved. When carrying out aggregation in several steps one may produce measures of fit not only for the entire unit of force but also for each subunit, assuming that it is possible to decide which objects belong to the same subunit.

More complete tests using data that correspond to real organizations should be carried out to evaluate the usefulness of the HMM technique for the aggregation problem before a real application is considered.

There exist some software packages for HMM modelling, e.g. HMMER and SAM. Both HMMER and SAM are designed for bioinformatics applications. One of these software packages might possibly be used for force aggregation purposes, although when performing this study we have developed our own software.

Acknowledgments

We acknowledge valuable remarks on an earlier version of this paper by our colleagues John Cantwell and Johan Schubert.

References

[1] P. Baldi and S. Brunak, *Bioinformatics - The Machine Learning Approach*. MIT Press, 1998.

[2] R. Durbin, S. Eddy, A. Krogh and G. Mitchison, *Biological sequence analysis*, Cambridge University Press, 1998.

[3] T. Koski, Lecture notes to a course on Hidden Markov Models, Dept. of Mathematics, Royal Institute of Technology, Stockholm, Sweden, 1999.

[4] J. Björnfot, Modeling the column problem. A study in information fusion, FOA-R--00-01448-505--SE (in Swedish), Defence Research Establishment, Linköping, Sweden, 2000.

[5] F.E. White, Managing data fusion systems in joint and coalition warfare, *Proceedings of EuroFusion98*, pp. 49–52, Great Malvern, UK, October 6–7, 1998.

[6] P. Husmann Rasmussen, Data Fusion Demonstrator, Projektstatus med udgangen af 1997, FOFT F-18/1998 (in Danish), Forsvarets forskningstjeneste, Copenhagen, Denmark, 1998.

[7] P. Gonsalves, G. Rinkus, S. Das and N. Ton, A hybrid artificial intelligence architecture for battlefield information fusion, *Proceedings of the Second International Conference on Information Fusion (Fusion ´99)*, pp. 463–468, Sunnyvale, California, July 6–8, 1999.

[8] J. Johnson, R. Chaney, Recursive composition inference for force aggregation, *Proceedings of the Second International Conference on Information Fusion (Fusion ´99)*, pp. 1187–1195, Sunnyvale, California, July 6–8, 1999.

[9] R.G. Cowell, A.P. Dawid, S.L. Lauritzen, D.J. Spiegelhalter, *Probabilistic Networks and Expert Systems*. Springer-Verlag, 1999.