A Hitchhiker's Guide towards Transactive Memory System Modeling in Small Group Interactions

ENZO TARTAGLIONE, University of Turin, Italy and LTCI, Télécom Paris, Institut polytechnique de Paris, France
BEATRICE BIANCARDI, LTCI, Télécom Paris, Institut polytechnique de Paris, France
MAURIZIO MANCINI, Sapienza University, Italy
GIOVANNA VARNI, LTCI, Télécom Paris, Institut polytechnique de Paris, France

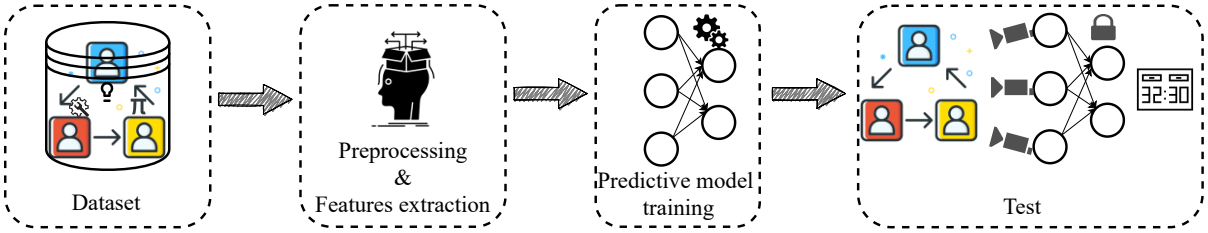

Fig. 1. Overview on the general structure for our approach.

Modeling Transactive Memory System (TMS) over time is an actual challenge of Human-Centered Computing. TMS is a group's meta-knowledge indicating the attribute of "who knows what". Conceiving and developing machines able to deal with TMS is a relevant step in the field of Hybrid Intelligence aiming at creating systems where human and artificial teammates cooperate in synergistic fashion. Recently, a TMS dataset has been proposed, where a number of audio and visual automated features and manual annotations are extracted taking inspiration from Social Sciences literature. Is it possible, on top of these, to model relationships between these engineered features and the TMS scores? In this work we first build and discuss a processing pipeline; then we propose four possible classifiers, two of which are artificial neural networks-based. We observe that the largest obstacle towards modeling the target relationships currently lies in the little data availability for training an automatic system. Our purpose, with this work, is to provide hints on how to avoid some common pitfalls to train these systems to learn TMS scores from audio/visual features.

CCS Concepts: • **Human-centered computing** → *Collaborative and social computing*; • **Computing methodologies** → *Artificial intelligence.*

Additional Key Words and Phrases: Multi-modal Group Behaviour Analysis, Social Signal Processing, Explainable Models, Transactive Memory System

ACM Reference Format:
Enzo Tartaglione, Beatrice Biancardi, Maurizio Mancini, and Giovanna Varni. 2021. A Hitchhiker's Guide towards Transactive Memory System Modeling in Small Group Interactions. In *Companion Publication of the 2021 International Conference on Multimodal Interaction (ICMI '21 Companion), October 18–22, 2021, Montréal, QC, Canada.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3461615.3485414

ICMI '21 Companion, October 18–22, 2021, Montréal, QC, Canada

1 INTRODUCTION

During interaction, teammates continuously share thoughts, affect and behaviors, contributing to the emergence of team phenomena referred to as "emergent states" [9, 14].

Three main categories of emergent states exist [7, 9]: the *cognitive* ones are related to the management of the team knowledge; the *behavioural* ones concern the activities and interactions between teammates; the *emotional* ones focus on teammates emotional responses. The Transactive Memory System (TMS) is a cognitive emergent state, consisting of a team meta-memory, combining each member's *personal* field of knowledge with the *awareness of each other's* ones [22]. Scholars in Social Sciences identify 3 TMS dimensions [10, 15]: (i) *Credibility*, the belief that the other team members' knowledge is correct and accurate; (ii) *Knowledge Specialisation*[1], the knowledge differentiation between teammates; (iii) *Coordination*, how much team members are capable of working together. The development of TMS follows 3 phases: *Encoding*, *Storage*, and *Retrieval*. During Encoding, the team members infer "who knows what", that is they identify the members' expertise. In Storage, they allocate the incoming information to team members matching a certain expertise [11]. Then, in Retrieval, they know from whom they can obtain the knowledge they need [11]. The 3 dimensions of TMS occur in all of these 3 phases. As previously showed in [2, 8, 9, 15], developing TMS in a team can significantly improve performance and productivity, reducing the individual cognitive load.

According to Clarkson *et al.* [4], it is becoming important to develop "computational artifacts in support of human endeavours". In this direction the role of machine is expected to shift from being just "idiots savants" [1] to becoming effective teammates, capable of engaging human partners in a team collaboration [13]. This is also the focus of the novel research field of Hybrid Intelligence (HI), aiming at "creating systems that operate as mixed teams, where humans and machines cooperate synergistically, proactively, and purposefully to achieve shared goals" [1]. In this perspective, we deem it is important that machines will be able to deal with TMS, i.e., to detect and, consequently, foster and leverage it while engaging human teammates in an interaction. Results in this direction could be useful, for example, in developing virtual agents able to adapt their behaviours according to the TMS level of a team.

While for decades this has been the object of research of social scientists, whose approach was mainly based on manual annotations and ratings from human observers, recent progresses in the automatic extraction of nonverbal features from audio-visual recordings allowed for applying automated techniques to infer emergent states in small group interactions (for a review, see [5]). Previous computational work on emergent states, however, focused on behavioural and emotional emergent states, as emergent leadership and cohesion, by neglecting the cognitive ones, such TMS. This could be explained by the fact that these states deal with more abstract concepts such as meta-memory, and so they are more difficult to be investigated by analysing nonverbal behaviours.

The problem of automated detection of TMS has been rarely addressed and discussed. In the era of "big data" and AI tools learning from examples, the typical little data availability in this context poses challenges in the predictive model's training. In this paper we propose a first investigation on computational models for TMS inference. In particular, our goal here is to provide a guideline, raising warnings on potential issues when modeling TMS's score dependency from audio-visual features and manual annotations. We observe that fully-automated deep learning (DL)-based approaches fail in small dataset scenarios, while a proper feature engineering, sided to traditional AI approaches like decision trees, provide the best results. This should not surprise us: features engineering provides a prior knowledge base which can not be learned and optimized by deep learning approaches from small data. Despite these obstacle, understanding data from the source, cleaning them from potential biases, engineering the extracted features, deploying a proper learning strategy to prevent overfit are the winning aspects we discuss and we provide insights on.

[1] from now on we refer to Knowledge Specialization as Specialization

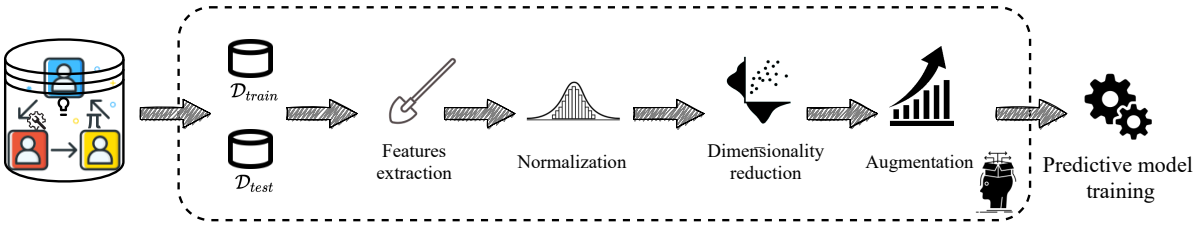

Fig. 2. Overview on potential action to be taken before training the classifier.

2 DATASET

The WoNoWa (**Wh**o k**No**ws **Wha**t) dataset is a multimodal (audio and video) dataset conceived to study TMS [3]. It is composed of 15 team interactions, each one involving 3 persons performing joint activities in a room. Self-assessment measures of the 3 dimensions of TMS at different times are also available. These assessments were collected through 5-point Likert items from well-estabilished questionnaires [10]. Although WoNoWa addresses all of the phases of TMS (i.e., *Encoding*, *Storage* and *Retrieval*), this work focuses only on the last phase, that is *Retrieval*. In the dataset, Retrieval was implemented as follows. First, each teammate was asked to choose which, among the following task, respectively related to a different field of expertise, they would like to be assigned to: (i) setting up a table by following the rules described in a video tutorial (Logistical expertise); (ii) computing conversions between the Imperial and the International System (Mathematical expertise); (iii) making origami (Manual expertise). Next, after the teammates accomplished the tasks they chose, they were asked to modify the setup of the table and to create an unknown origami, this time following a list of dimensions (given in the Imperial system). Finally, and this is the step on which we focus in the paper, they were free to self-assign the same 3 tasks in any way they wanted (but they could not chose the one they just performed). So, the teammates needed each other's expertise, resulting in collaboration and interaction between them. We decided to focus on this step because here the team is expected to have developed TMS through the previous steps. This was confirmed by the higher self-assessed scores of *Specialisation* and *Coordination*, as well as higher scores of *Credibility* that teammates reported in this step with respect to those ones rated in the previous steps.

3 PROPOSED FRAMEWORK

Considering the peculiar nature of the data, in our study we investigate over a number of possibilities to infer the Specialization, Credibility and Coordination self-assessed scores. Towards this end, we may stage the proposed approaches in the following phases (see Figure 2) : i) features extraction ii) features normalization; iii) dimensionality reduction; iv) data augmentation and v) training of a model. In the next sections we will discuss more in depth each of these stages.

3.1 Train/test split

Managing a proper train/test split is, in certain contexts, an overlooked common pitfall [6, 19]. When splitting a dataset $\mathcal{D}$ in train $\mathcal{D}_{train}$ and test $\mathcal{D}_{test}$ sets, we should ask ourselves whether there is some unwanted data leakage between the two sets. Let us assume we work with labeled data, and we consider data belonging to the same μ-th label. Evidently, we know that, *ideally*, within X^{μ}_{train} and X^{μ}_{test} we have a mutual information:

$$I(\mathcal{D}^{\mu}) = I^{\mu} \tag{1}$$

where I^{μ} is the information we target to train some classification model to extract (normalized in range $[0; 1]$). In order to guarantee no information leakage between train and test set, we should guarantee that:

$$I\left(x_i^{\mu}, t_j^{\mu}\right) \leq I^{\mu}, \forall x_i^{\mu} \in \mathcal{D}_{train}, t_j^{\mu} \in \mathcal{D}_{test} \tag{2}$$

Dramatically, there are real cases in which there exist some contexts in which for random dataset splits there are some examples where $I\left(x_i^{\mu}, t_j^{\mu}\right) \to 1$.

Looking at the dataset, to avoid overestimation for the generalization performance, we perform the train/test split on *groups* rather than on single participant. Towards this end, we are going to study, in the remainder of this section, data extracted from 13 groups. The 2 remaining groups are isolated and will be used at inference time only, when testing the generalization performance of the evaluated models.

3.2 Features extraction

Table 1 reports the automatically and manually extracted features of the teammates' nonverbal behaviour. These features concern audio, movement and spatial arrangements of the teammates in the psychical space. For the sake of clarity, the Personal Area is the location in the room in which each teammate worked on the assigned task, while the Others Area corresponds to the locations in which the other teammates worked on their assigned tasks. Finally, the Common Area identifies the remaining areas of the room.

In this study, we used features both at *low granularity*, that is computed over the entire third step of the Retrieval phase, and at *high granularity*, that is computed over short time windows (15s, with 3s of overlap). High granularity was applied to investigate the possibility of automatically learn to extract the features which represent the best-fit to infer TMS scores. *QoM, QoME, HV, HVE* and *HDir* are the features that were also computed at high granularity (underlined in Table 1).

3.3 Features normalization

After features have been extracted, typically a good practice involves their normalization. Data features are typically affected by biases: if we are able to remove these prior the learning algorithm is going to build relationships between features, we allow more complexity in the AI algorithm to be deployed, boosting the generalization performance. There are some learning algorithms, like decision trees, which are insensitive to data normalization by design.

Figure 3 displays the average value and the standard deviation for the features extracted at low-granularity.

Despite the majority of the extracted features' average value is in range $[10^1; 10^2]$, a subset of features (*TST, TSL, ASL, TAI, TSI*) shows typically low values, certified by a low standard deviation. Considering the significant scale difference between these quantities, most of the typically deployed learning approaches will be biased towards focusing on the features having a larger average magnitude.

Focusing on the evaluated standard deviations, we observe that just a very restricted number of features show high values. However, for these features (namely the average time spent having triangular f-formations), it is very important to properly scale their very large and typical variation.

3.4 Target classes unbalance and considerations on training strategies

After we have taken into account issues more related to data input, we can also have a more in-depth look at the distribution for the target classes. Looking at the full dataset's TMS self-assessed scores (Figure 4), we observe a remarked unbalance between the various scores. In particular, both Credibility and Coordination are heavily biased towards "score 4", while Specialization is somewhat balanced between "score 4" and "score 5". In general, very little samples are good representatives for different scores, namely "score 2" and "score 3", while samples

Table 1. Description of the features computed for the analyses.

Name	Feature	Description
Head Distance	*HDist*, *HDistSD*	average distance of each team member's head from the other 2 team members
Performance	*Perf*	a numerical score quantifying the goodness of the task performance
Common Area Occupation Frequency	*CAF*	number of times teammates occupy the Common Area
Common Area Occupation Percentage	*CAP*	percentage of time spent by teammates in the Common Area
Common Area Occupation Mean Time	*CAT*	average time spent by teammates in the Common Area
Others' Area Occupation Frequency	*OAF*	number of times teammates worked in the Others Area
Others' Area Occupation Percentage	*OAP*	percentage of time spent by teammates in the Others Area
Others' Area Occupation Mean Time	*OAT*	average time spent by teammates in the Others Area
Personal Area Occupation Frequency	*PAF*	number of times teammates worked in the Personal Area
Personal Area Occupation Percentage	*PAP*	percentage of time spent by teammates in the Personal Area
Personal Area Occupation Mean Time	*PAT*	average time spent by teammates in the Personal Area
Other f-formations Frequency	*OthffF*	frequency of "Other" (i.e., L-shape or Side-by-side) f-formations
Other f-formations Percentage	*OthffP*	time percentage spent in "Other" f-formations
Other f-formations Mean Time	*OthffT*	average duration time spent in "Other" f-formations
Triangular f-formations Frequency	*TrffF*	number of episodes of Triangular arrangements per minute
Triangular f-formations Percentage	*TrffP*	percentage of time in which the group was engaged in a Triangular F-formation during the task
Triangular f-formations Mean Time	*TrffT*	average duration of an episode in which the group remained in a Triangular arrangement without changing
Semi-Circular f-formations Frequency	*SCffF*	number of episodes of Semi-circular arrangements per minute
Semi-Circular f-formations Percentage	*SCffP*	percentage of time in which the group was engaged in a Semi-circular F-formation during the task
Semi-Circular f-formations Mean Time	*SCffT*	average duration of an episode in which the group remained in a Semi-circular arrangement without changing
Head Velocity	*HV*, *HVSD*	magnitude of the 1st derivative of the head position
Head Velocity Entropy	*HVE*, *HVESD*	Sample Entropy of Head Velocity
Quantity of Motion	*QoM*, *QoMSD*	frame differencing between 2 consecutive person's silhouettes
Quantity of Motion Entropy	*QoME*, *QoMESD*	Sample Entropy of Quantity of Motion
Head Directness	*HDir*, *HDirSD*	how much direct vs indirect the person's head trajectory is
Total Speaking Turns	*TST*	number of speaking turns, normalised by the interaction length
Total Speaking Length	*TSL*	total speaking time, normalised by the interaction length
Average Speaking Turn	*AST*	average speaking turn duration
Total Attempted Interruptions	*TAI*	number of attempted interruptions, normalised by the interaction length
Total Successful Interruptions	*TSI*	number of successful interruptions, normalised by the interaction length
Successful Interruptions Percentage	*SIP*	percentage of successful over attempted interruptions

with "score 1" associated are completely missing, making it impossible to both train a model and test it for such a class.

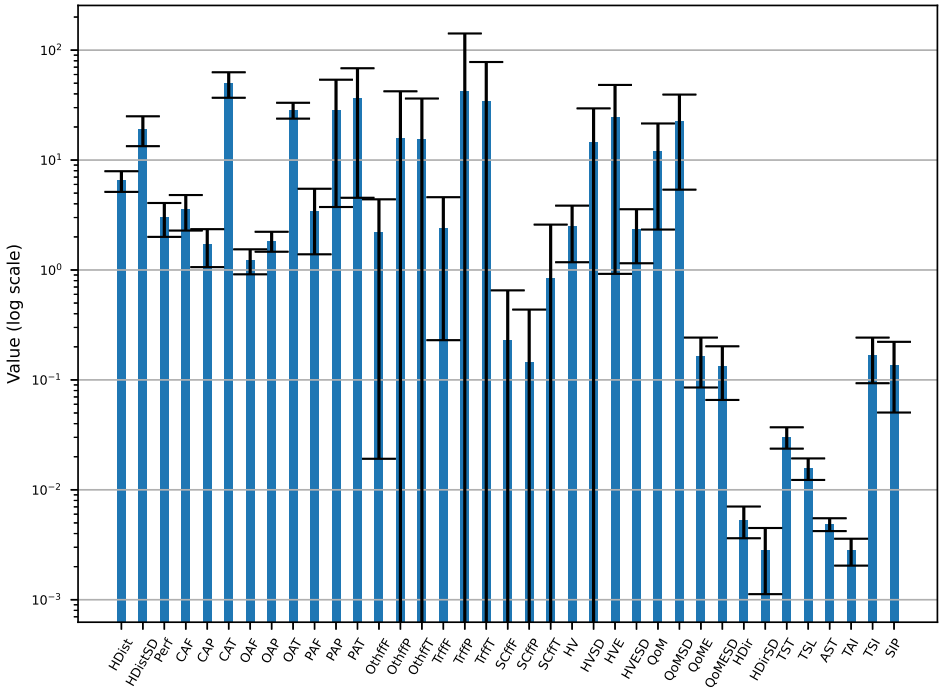

Fig. 3. Distribution on the extracted features from the dataset.

There are standard approaches which can mitigate the data unbalance problem at train time. When training an artificial neural network-based model, when training happens in an iterative fashion minimizing some loss function L evaluated on the training set X_{train}, to prevent target unbalance to heavily bias the model, it is good practice to i) use a weighted sampler to build a minibatch with class balancing and ii) to re-weight the loss contribution for each class. For both the approaches, we are required to compute, for each μ-th class, a weight:

$$w^{\mu} = \begin{cases} 0 & if \ \|\mathcal{D}^{\mu}\|_0 = 0 \\ \frac{\|\mathcal{D}\|_0}{\|\mathcal{D}^{\mu}\|_0} & otherwise \\ \frac{\|\mathcal{D}\|_0}{\sum_c \|\mathcal{D}^{c}\|_0} & \end{cases} \tag{3}$$

where $\|\cdot\|_0$ denotes the ℓ_0 norm, or the cardinality of the set. When the target function is not necessarily a differentiable proxy for the target function (like in artificial neural network models), it is certainly necessary to use a balanced metric in order to evaluate the accuracy over the given model. Towards this end, we are going to use the *balanced accuracy* as our target. Let us name our model's prediction for the i-th sample y_i: we first compute the accuracy for a specific μ-th score (from 1 to 5) of the currently evaluated TMS score:

$$A^{\mu} = \begin{cases} 0 & if \ \exists \ y_i \mid \|\mathcal{D}^{\mu}\|_0 = 0 \wedge y_i = t \\ 1 & if \ \|\mathcal{D}^{\mu}\|_0 = 0 \wedge y_i \neq t \forall i \\ \frac{1}{\|\mathcal{D}^{\mu}\|_0} \sum_i \delta_{\mu, y_i} & otherwise, \end{cases} \tag{4}$$

where $\delta_{a,b}$ is the Kronecker delta. Then, to compute the balanced accuracy, we average the performance measured for each class. Here on, we will shortly refer to accuracy as the average of (4) over all the classes.

3.5 Dimensionality reduction

Dimensionality reduction transforms data from a (sparse) high-dimensional space into a (dense) low-dimensional space. The main purpose of these approaches is to retain the meaningful properties of the original data, on top of which pattern mining algorithms attempt to model relationships to proper infer the target classification. Directly working in high-dimensional spaces can be undesirable for many reasons; raw data are often sparse as a consequence of the curse of dimensionality, and analyzing the data is usually computationally intractable. Typical DL-based approaches employ architectures which can be roughly divided in two stages:

- *feature extraction*, in which data are compressed in a small hidden latent space and their dimensionality is hereby reduced;
- *classification*, in which the output of the features extraction stage is processed towards the learning goal (in our case, classification).

Hence, dimensionality reduction is typically, implicitly, automatically deployed (and learned) when using DL-based approaches (hence, when we train with high granularity). However, when dealing with other approaches, it is important to employ a strategy aiming at reducing the data dimensionality towards the learning's success (low granularity). Towards this end, many approaches have been proposed in the literature, adopting evolutions of the DBSCAN algorithm [18], proposing hyperspectral dimensionality reduction [16] or training autoencoders [12, 17, 21].

Unfortunately, most of the above-mentioned works are not applicable in our context, given the reduced size of the dataset we have decided to work with. For this reason, more traditional and general approaches are the best fits for our case. Towards this end, Principal Component Analysis (PCA) is the elected approach. A high-level analysis on the extracted features is reported in the following. Merely for display purposes, in Section 3.7 we will employ t-SNE [20], which qualitatively provides good visualization on data clustering.

3.6 Discussion on the extracted features

Figure 5 shows the (normalized) cumulative variance achieved when extracting the top k components. As observed in Section 3.3, without a proper feature normalization at pre-processing, the features *apparently* containing most of the information in the data are underestimated. For example, in Figure 5, setting as cut-off threshold the 95% over the total variance in the data, we find for un-normalized data that 5 components are sufficiently representative for the dataset. However, according the analysis on the dataset performed in Section 3.3, there is a subset of features with a dominant variance, dictated also by the different scale of the data.

Let us have a closer look at Figure 6a, in which we display the PCA loading for the first 17 extracted components (95% of the total variance). Interestingly, we observe some clusters which are automatically extracted by the PCA. For example, in the first component the three semi-circular f-formation features have a similar negative loading, while other f-formations have a similar positive loading. This was expected, as we know that there is an intrinsic relationship between these features, but it is interesting to observe how these groups of features are mutually dependent. Towards this end, we also provide the average PCA loading given the target TMS score, as shown in Figure 6b, Figure 6c and Figure 6d. From this visual representation, more insights rise: for example, a low coordination score positively correlates to a high quantity of motion's entropy (*QoME*), which is a symptom of highly chaotic movements, or a high credibility correlates with a low quantity of motion (*QoM*), meaning that the least the person stands in a given position, the highest its credibility. Looking at the specialization scores, we can deduct that low specialization scores correlate negatively with time spent in semicircular formations and in a low number of speak turns (*TST*).

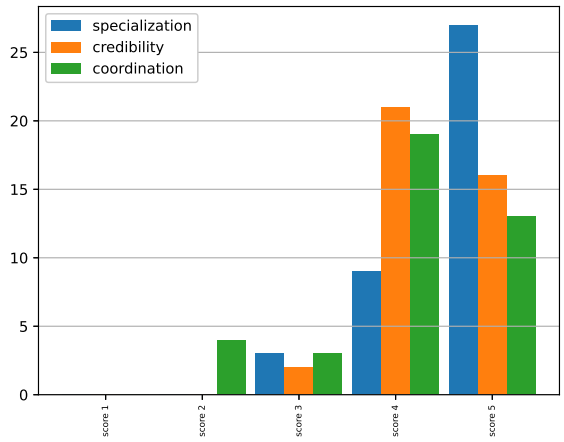

Fig. 4. Distribution of the target scores in the dataset.

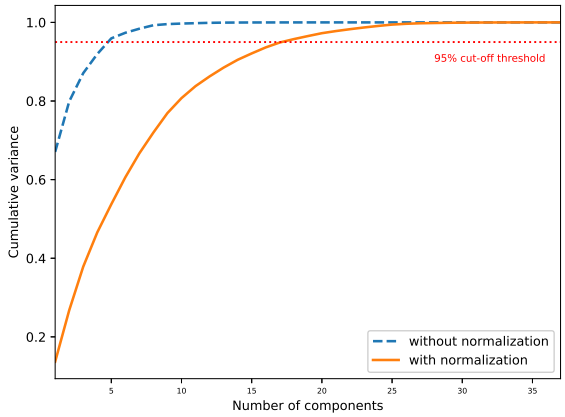

Fig. 5. Cumulative variance of the first k components of the PCA evaluated on un-normalized and on normalized data.

Training a classification model on top of these features attempts to properly model all of these relationships in an automated way, simply looking at the data and forcing the input-output relationship. However, the low data availability constitutes a significant obstacle towards such a goal. In order to tentatively overcome this problem, we can augment the train data, as explained in the following subsection.

3.7 Data augmentation

In small data scenarios, it is typically a good strategy to employ a data augmentation strategy. Despite the ideal scenario depicted in Section 3.1, there are cases, out from our control, where:

$$I\left(\mathcal{D}^{\mu}_{train}\right) \geq I^{\mu} \tag{5}$$

as there might be spurious correlations between different samples. This, unfortunately, might lead to a deterioration of the generalization performance of the trained model. To alleviate this problem, data augmentation strategies can be designed. Their goal is twofold: i) remove spurious correlations between training data, and ii) increase the size of the training set. More formally, we can say that that we can apply a certain sequence of transformations $\mathcal{T}(\cdot)$ to the training set $\mathcal{D}_{train}$ such that:

$$I\left[\mathcal{T}\left(\mathcal{D}^{\mu}_{train}\right)\right] \rightarrow I^{\mu} \tag{6}$$

Typically, standard data augmentation strategies for image processing include rigid transformations or brightness/contrast/hue shift. Considering the nature of the features we are working with, we apply, after the feature normalization stage as described in Section 3.3, a Gaussian noise with standard deviation 0.1, increasing the size of the training set by a factor 100×.

Figure 7 displays the 3D projection for the augmented normalized features (after PCA cutoff to 95% of the variance is applied), for the three target classes to predict (specialization, credibility, coordination). For visualization purposes, the 3D projection is performed using t-SNE [20]. Qualitatively, we observe some clusters around the different scores: in the next section we are going to compare classification performance between four different classification approaches, deploying at the same time an ablation study on the data processing strategies proposed.

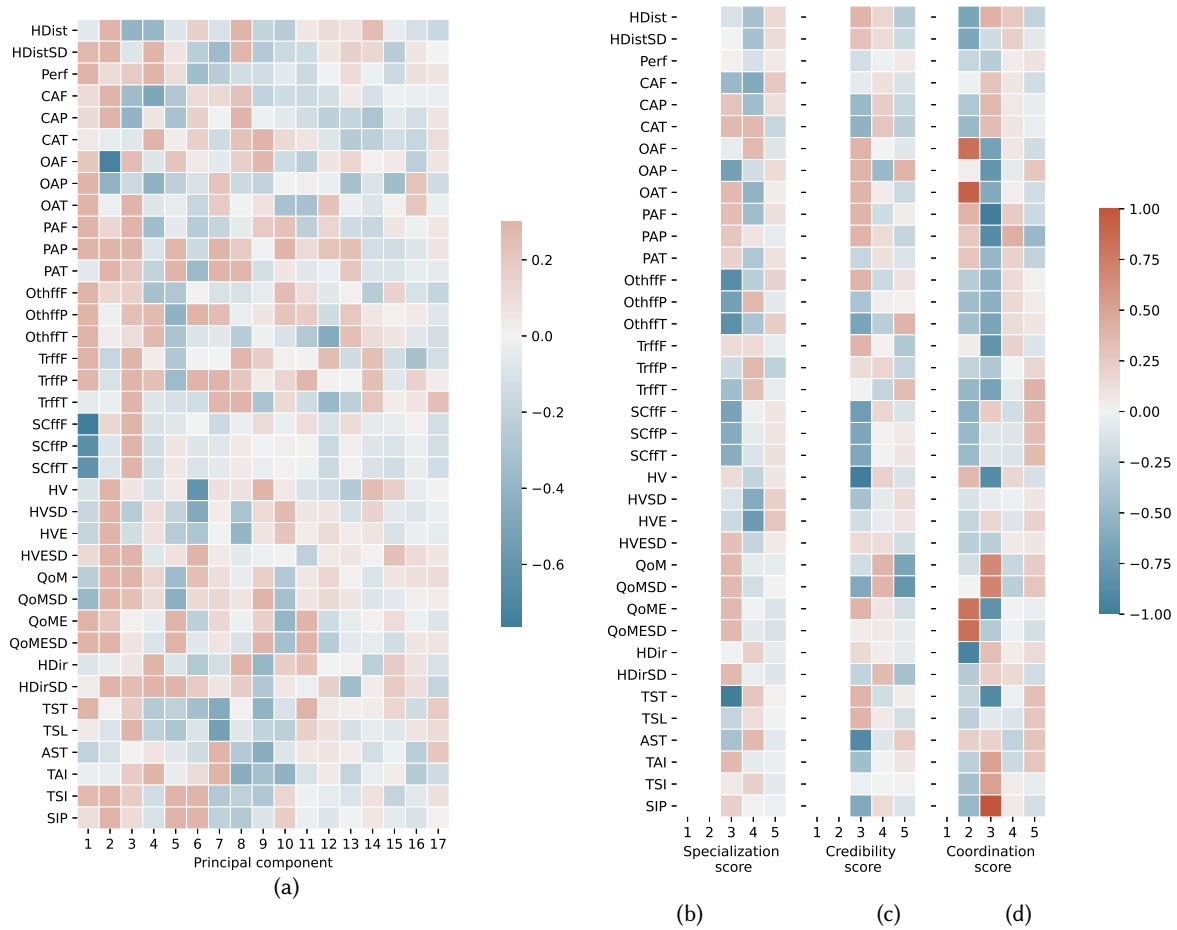

Fig. 6. PCA loading for the first 17 components (95% of cumulative variance) (a) and divided per target class (b-d). White columns are classes missing.

4 EVALUATION

In this section we are going to discuss the results obtained training some of the most common classification algorithms employing the learning design choices discussed in Section 3.

4.1 Implementation

Convolutional neural network model (CNN). It is an artificial neural network model working with high granularity features. It is designed to receive as input features from 5 different channels (*QoM, QoME, HV, HVE* and *HDir*, as discussed in Section 3.2). This model has a feature extractor made of two 1D convolutional layers of size 10 and 20, each having kernel of size 5×1, and data are processed with stride 1. The classification is entirely processed in the output layer, which is trained to provide the three TMS scores. The learning process minimizes

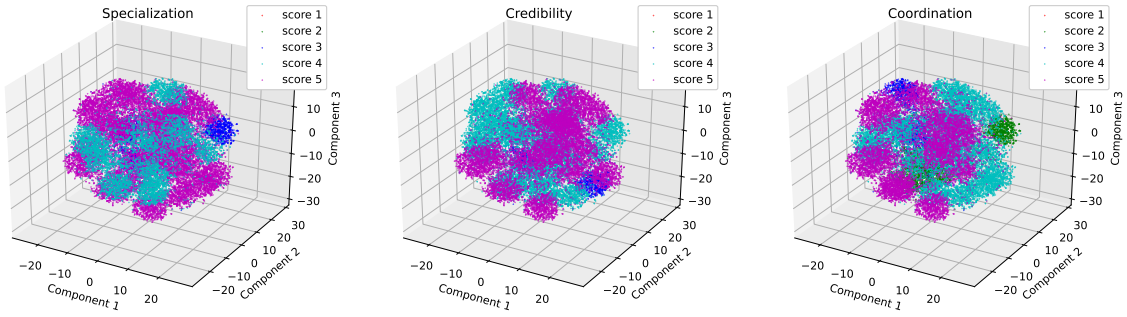

Fig. 7. Projection of the features extracted from the dataset with low granularity, after feature normalization, dimensionality reduction (95% variance) and projection in a 3D space: specialization scores (a), credibility scores (b) and coordination scores (c).

a categorical cross-entropy on the output, which is optimized using SGD with learning rate 10^{-2}, batch size 40, momentum 0.9, weight decay 10^{-5} for 50 epochs.

Multi layer perceptron (MLP). It is an artificial neural network model working with low granularity features. It is designed to receive all the 37 possible low-granularity features and provide the TMS scores as output. It is a three-layers ANN model, with two hidden fully-connected layers having size 100 and 10, ReLU-activated. Likewise for the CNN training, a categorical cross-entropy is here minimized using SGD with learning rate 10^{-2}, batch size 100, momentum 0.9, weight decay 10^{-5} for 100 epochs.

Logistic regression model (LR). The logistic regression model is trained on low granularity features. In this case, we train three distinct models (one for each of the TMS scores), and also in this case we target the categorical cross-entropy loss minimization. We optimize the model with LBFGS with a ℓ_2 penalty term, having 500 as wall epochs.

Decision Tree (DT). We also employ this non-parametric supervised learning method. The limit of this approach lies in the fact that the learnable relationships in the data are piecewise constant approximation, which limits the capability of learning complex non-linear relationships, making these models a good fit for small datasets. We train DT on low granularity features, using the CART algorithm and gini index as split criterion, the maximum tree depth has been set to 10.

Additional details. All training and inference algorithms are implemented in Python 3.8. We have trained four different classification models. The code for CNN and MLP is written using `pytorch 1.8` and the learning is performed using a NVIDIA RTX2070 GPU equipped with 8GB RAM, while for LR and DT the `sklearn 0.24.2` library has been used and training has been performed on an Intel i7 9900K CPU equipped with 32GB RAM. Features normalization, data augmentation and dimensionality reduction (PCA with 95% variance extraction) have been implemented using both `numpy 1.21` and `sklearn 0.24.2`.[2]

[2]The source code is available at https://github.com/enzotarta/A-Hitchhiker-s-Guide-towards-Transactive-Memory-System-Modeling-in-Small-Group-Interactions

Table 2. Summary of the experimental configurations (with different options for the pre-processing pipeline: features normalization - Feat norm, dimensionality reduction - Dim red, data augmentation - Data aug), and results.

Features granularity	**Feat norm**	**Dim red**	**Data aug**	**Trained model**	**Performance (test) [%]** Specialization	Credibility	Coordination
-	-	-	-	Random	20.0	20.0	20.0
High	✓	✗	✓	CNN	46.8 ± 20.9	46.5 ± 20.3	25.7 ± 16.6
High	✓	✗	✗	CNN	43.3 ± 10.1	38.9 ± 10.8	16.7 ± 8.6
Low	✓	✗	✗	MLP	37.0 ± 17.0	55.6 ± 12.0	27.6 ± 17.1
Low	✓	✗	✓	MLP	30.8 ± 3.8	28.8 ± 5.7	50.0∗
Low	✓	✗	✗	LR	25.0∗	25.0∗	27.5∗
Low	✓	✓	✗	LR	43.3∗	40.0∗	33.3∗
Low	✓	✓	✓	LR	50.5∗	51.7∗	58.3∗
Low	✗	✗	✗	DT	45.7 ± 7.3	57.1 ± 2.6	59.1 ± 7.2
Low	✓	✗	✗	DT	44.8 ± 7.7	41.2 ± 7.3	59.2 ± 6.8
Low	✗	✗	✓	DT	42.8 ± 3.8	54.2 ± 2.6	80.0∗
Low	✓	✗	✓	DT	42.8 ± 3.8	54.2 ± 2.6	80.0 ± 1.1
Low	✓	✓	✓	DT	**51.8 ± 2.1**	**58.3**∗	**83.3**∗

4.2 Discussion on the results

A summary on the TMS's classification results is provided in Table 2. Please notice that the performance measures are weighted accuracies, following (4). The performance is evaluated on the same test set, following the criteria discussed in Sec. 3.1, and the results are averaged over 100 seeds. We indicate with "∗" the results for which the standard deviation is lower than 0.1%.

First, we observe that the CNN model, on this dataset, fails in achieving good generalization. Despite the big potentialities these models have, given that they are able to model complex non-linear relationships between data, the parameters to be learned (and tuned) are approximately 10k. Despite regularization strategy deployed (like weight-decay) and data augmentation, the limited size of the dataset makes the learning of these models harder than learning shallower ones, and learning an automatic features engineering in this scenario is more than chimeric.

Deploying low granularity features, extracted using a prior knowledge pool, when working on small data seems to be the way to move on. Comparing MLP and LR, we observe that the best performance is achieved when training a LR model with PCA deployed to reduce the data dimensionality and data augmentation. Differently from MLP, which also attempts to learn an optimal dimensionality reduction with the hidden layers, the LR model working directly on the PCAd features achieves a better performance. However, the overall best performance is achieved with DT models. This gain in generalization performance can be explained from the fact DT models simple piecewise linear approximation, which generalizes better than training non-linear models on small data, which are more prone to overfit data.

One main advantage of DT models is given by their interpretability. As an example, in the configuration with low features granularity and the complete preprocessing pipeline (last row in Table 2), we can have a look at the

DTs having the median performance on the test set.[3] As intuitive, we observe diversity in the features selected for each of the TMS scores, in particular, the selected features for:

- Specialization refer to the average occupation time for the personal (PAT) and others area (OAT);
- Credibility refer to percentages of occupation for personal (PAP), other people's (OAP) and common area (CAP);
- Coordination refer to average time spent in semi-circular f-formation (SCffT), Quantity of Motion (QoM), Quantity of Motion entropy (QoME), occupation frequency of others area (OAF) and average occupation time for common area (CAT).

Interestingly, we observe that audio features like TSL (total speaking length) are used for more scores, denoting a more general role in TMS scores modeling. For example, we qualitatively observe that low TSL is associated to high scores in Specialization, Credibility and Coordination. The structure of the DT models is complex and given the current performance, we can not fully trust them. However, considering that random guess is 20% and we are above 50%, we expect having more data and holding the discussed setup will be decisive in enhancing the model's performance.

4.3 Conclusion and future work

In this work we shed some light over the possibility that machines will be able to deal with TMS, i.e., to detect it while engaging human teammates in an interaction. More specifically, we trained classifiers to infer TMS self-assessed scores in small groups interactions. Given the complexities around the data collection process for this specific context, differently from some of the most known contexts where AI-based algorithms are trained, we are in a "small data" scenario, where training from examples is a more delicate and complex issue to address. We have hereby discussed some of the most common approaches and possible sub-optimalities in this specific context. We have shown that a proper data pre-processing stage, including data normalization and dimensionality reduction, coupled with data augmentation provides the most promising approach, achieving the best performance. Even more importantly, DL-based approaches show to be harder to optimize when compared to decision trees: despite their broadly glorified potentiality of learning complex non-linear relationships between data, they are extremely prone to overfit data, and in small data scenarios, without a proper prior knowledge base, its hyper-parameters are extremely hard to optimize. When more data will be available, we believe DL approaches will outperform other techniques; however, when working with a low mole of data, simpler (i.e., with less parameters to learn) and more "white box" models, like PCA coupled with, for example, DT, should be favored.

REFERENCES

[1] Zeynep Akata, Dan Balliet, Maarten De Rijke, Frank Dignum, Virginia Dignum, Guszti Eiben, Antske Fokkens, Davide Grossi, Koen Hindriks, Holger Hoos, et al. 2020. A Research Agenda for Hybrid Intelligence: Augmenting Human Intellect With Collaborative, Adaptive, Responsible, and Explainable Artificial Intelligence. *Computer* 53, 8 (2020), 18–28.

[2] John R Austin. 2003. Transactive memory in organizational groups: the effects of content, consensus, specialization, and accuracy on group performance. *Journal of applied psychology* 88, 5 (2003), 866.

[3] Beatrice Biancardi, Lou Maisonnave-Couterou, Pierrick Renault, Brian Ravenet, Maurizio Mancini, and Giovanna Varni. 2020. The WoNoWa Dataset: Investigating the Transactive Memory System in Small Group Interactions. In *Proceedings of the 2020 International Conference on Multimodal Interaction*. 528–537.

[4] Edward Clarkson, Jason A Day, and James D Foley. 2006. An educational digital library for human-centered computing. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*. 646–651.

[5] Daniel Gatica-Perez. 2009. Automatic nonverbal analysis of social interaction in small groups: A review. *Image and vision computing* 27, 12 (2009), 1775–1787.

[3] Available at https://github.com/enzotarta/A-Hitchhiker-s-Guide-towards-Transactive-Memory-System-Modeling-in-Small-Group-Interactions/tree/main/decision_tree_examples

[6] Mahesh Gour, Sweta Jain, and T Sunil Kumar. 2020. Residual learning based CNN for breast cancer histopathological image classification. *International Journal of Imaging Systems and Technology* 30, 3 (2020), 621–635.

[7] Rebecca Grossman, Sarit B Friedman, and Suman Kalra. 2017. Teamwork processes and emergent states. *The Wiley Blackwell handbook of the psychology of team working and collaborative processes* (2017), 243–269.

[8] Andrea B Hollingshead. 2000. Perceptions of expertise and transactive memory in work relationships. *Group processes & Intergroup relations* 3, 3 (2000), 257–267.

[9] Steve WJ Kozlowski and Daniel R Ilgen. 2006. Enhancing the effectiveness of work groups and teams. *Psychological science in the public interest* 7, 3 (2006), 77–124.

[10] Kyle Lewis. 2003. Measuring transactive memory systems in the field: scale development and validation. *Journal of applied psychology* 88, 4 (2003), 587–604.

[11] Jenny Liao, Nerina L Jimmieson, Anne T O'Brien, and Simon LD Restubog. 2012. Developing transactive memory systems: Theoretical contributions from a social identity perspective. *Group & organization management* 37, 2 (2012), 204–240.

[12] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* (2015).

[13] Thomas W Malone. 2018. How human-computer'Superminds' are redefining the future of work. *MIT Sloan Management Review* 59, 4 (2018), 34–41.

[14] Michelle A Marks, John E Mathieu, and Stephen J Zaccaro. 2001. A temporally based framework and taxonomy of team processes. *Academy of management review* 26, 3 (2001), 356–376.

[15] Richard L Moreland and Larissa Myaskovsky. 2000. Exploring the performance benefits of group training: Transactive memory or improved communication? *Organizational behavior and human decision processes* 82, 1 (2000), 117–133.

[16] Juan Pablo Rivera-Caicedo, Jochem Verrelst, Jordi Muñoz-Marí, Gustau Camps-Valls, and José Moreno. 2017. Hyperspectral dimensionality reduction for biophysical variable statistical retrieval. *ISPRS journal of photogrammetry and remote sensing* 132 (2017), 88–101.

[17] Rajeev Sahay, Rehana Mahfuz, and Aly El Gamal. 2019. Combatting adversarial attacks through denoising and dimensionality reduction: A cascaded autoencoder approach. In *2019 53rd Annual conference on information sciences and systems (CISS)*. IEEE, 1–6.

[18] Erich Schubert, Jörg Sander, Martin Ester, Hans Peter Kriegel, and Xiaowei Xu. 2017. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. *ACM Transactions on Database Systems (TODS)* 42, 3 (2017), 1–21.

[19] Enzo Tartaglione, Carlo Alberto Barbano, Claudio Berzovini, Marco Calandri, and Marco Grangetto. 2020. Unveiling covid-19 from chest x-ray with deep learning: a hurdles race with small data. *International Journal of Environmental Research and Public Health* 17, 18 (2020), 6933.

[20] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[21] Yasi Wang, Hongxun Yao, and Sicheng Zhao. 2016. Auto-encoder based dimensionality reduction. *Neurocomputing* 184 (2016), 232–242.

[22] Daniel M Wegner. 1987. Transactive memory: A contemporary analysis of the group mind. In *Theories of group behavior*. Springer, 185–208.