Communication in Soccer Simulation: On the Use of Wiretapping Opponent Teams

Thomas Gabel, Philipp Klöppner, Eicke Godehardt, Alaa Tharwat

Machine Learning and Computational Intelligence Group (MLCI)
Frankfurt University of Applied Sciences
60318 Frankfurt am Main, Germany
{tgabel|godehardt|aothman}@fb2.fra-uas.de, kloeppne@stud.fra-uas.de

Abstract. Inter-agent communication has been playing an important role in soccer simulation 2D since its introduction. Its primary usage has been to communicate with teammates in order to share state observations to fill gaps in the players' world models, to announce near future actions like passes or requesting passes, as well as for sharing and synchronizing on locker room agreements. In this paper, by contrast, our focus is on the communication of the opponent team. We present an approach for wiretapping and decoding opponent communication and systematically evaluate its impact. Our main finding is that a team that wiretaps its opponent and exploits intercepted information appropriately, can boost its own playing performance significantly.

1 Introduction

When multiple agents need to act independently of one another, under real-time constraints, and under a partial view of the world, inter-agent communication is one mean to mitigate the challenges of distributed decision-making and of coordinating agent behaviors. In robotic soccer, we face all of these challenges, with different nuances across leagues. Given that robotic soccer represents also a highly competitive domain, it is standing to reason that nearly any team exploits the granted possibilities of communication to the extent the rules of the respective RoboCup league permit. As a consequence, the question arises whether some team might gain an advantage, if it decides to wiretap its opponent and if it – assuming that it somehow understands the contents of foreign communication – exploits such eavesdropped information during its own decision-making process. This is the research question that we are going to explore for the 2D simulation league, subsequently.

We start off by providing relevant background information on the mechanics and the general role of communication in soccer simulation (Section 2), before we present the core ideas of an approach to learn the meaning of opponent communication in Section 3 [4]. The task will be cast as a supervised learning problem where deep convolutional neural networks will do the actual work of decoding some foreign language message to readable data. Since the belonging implementation has been realized using the TensorFlow (TF) framework

[1], we advocate Section 4 to a brief review of software engineering challenges encountered when incorporating the TF-based implementation into an existing RoboCup team. In Section 5, we return to our research question and empirically evaluate the quantitative impact of our approach. In so doing, we introduce various communication-related modifications to our team FRA-UNIted, including the aforementioned wiretapping ideas, and assess its performance against the current world champion (Helios [2]). Besides reporting and visualizing the remarkable impact, we critically discuss the approach before we conclude (Sec. 6).

2 Background

In what follows, we outline the general mechanics of communication in the 2D simulation league and discuss relevant related work.

2.1 Communication in RoboCup's Soccer Simulation 2D League

While direct communication among agents is prohibited, each agent is allowed to broadcast a string of up to 10 characters in each of the 6000 time steps that a regular game is made of. Such say messages are received by the Soccer Server [9] and are handed on to those players in the subsequent simulation cycle which had put their listening attention to the sender. Each agent can maximally hear one message from a teammate plus one message from an opponent at a time.

```
9,0  Recv HELIOS2017_10: (kick 86.8 63.1)(turn_neck -73)(attentionto our 11)(say "BybzQQ(u9U")
9,0  Recv FRA-UNIted_5: (dash 100 -0)(turn_neck 7.65998)(say "72gtaUuDWT")(attentionto our 4)
9,0  Recv HELIOS2017_2: (turn -53.967)(turn_neck -31)
9,0  Recv HELIOS2017_4: (dash 60)(turn_neck -1)
9,0  Recv FRA-UNIted_4: (dash 100 -0)(turn_neck -89.2104)(say "6zfuDWN9VT")(attentionto our 6)
9,0  Recv HELIOS2017_11: (dash 100)(turn_neck 0)(say "PzdAc_")
```

Fig. 1. Excerpt of the text log file of RoboCup 2017's final match (shortened).

Figure 1 shows an excerpt of the text log file of the 2017 final match, which among other things reveals which actions were taken by some of the players during time step 9. Most interestingly, some of the players made use of communication by issuing say messages of up to 10 chars length, though, from a human perspective, it seems nearly impossible to understand the contents of these strings. In this regard, we refer to Section 5.2, where we exemplarily and empirically reason on the actual contents of such say messages.

2.2 Related Work

Communication is an active field of research in the multi-agent systems community and, specifically, in robotic soccer. There has been a lot of work on communication across leagues (e.g. in the MidSize league [8] or in the 3D simulation league [7]). For the 2D league, communication has been an important

building block ever since, though the rules on how to communicate, as enforced by the Soccer Server [9], have changed over the years. In early work, Stone and Veloso [12] focused on developing techniques for inter-agent communication in unreliable, low-bandwidth environments, assuming that agents can communicate 256 bytes every two time steps. Starting with the change from Soccer Server version 7 to 8 in 2002, the maximum length of players' say messages has, however, been limited severely (to the 10 chars mentioned), rendering "plain text" communication nearly useless. The general potential for communication to improve distributed decision-making in multi-agent systems is also considered in [13]. Our experimental methodology is, to some extent, in line with [10], who theoretically and empirically evaluated the utility of varying communication protocols in soccer simulation, and with [15] to which we relate our work more thoroughly in Section 5. Finally, we point the reader to the related work section in [15] for an excellent overview on multi-agent communication and coordination.

3 Eavesdropping Opponent Agent Communication

In this section, we aim at providing a concise overview of the approach to eavesdrop and decode intercepted messages sent by an opponent soccer simulation team, which has first been presented in [4]. The basic idea of this approach is to pose the problem as a supervised learning task and to leverage state-of-the-art deep learning techniques for recognizing the meaning of messages heard. The authors make the obvious assumption that the contents of an intercepted message bears information whose transmission is beneficial to the opponent team, containing match-related data such as (1) ball-related, (2) player-related, (3) pass-related, or (4) team strategy-related junks of information. While the authors of [4] have primarily concentrated on (1) and (3), the focus of the paper at hand is on pass-related information, more specifically on the recognition of adversarial pass announcements, as well as on the quantitative impact that wiretapping the opponent agents can have on our own team's playing performance.

3.1 Learning Problem Formalization

Defining the problem as a supervised learning task, a set of training patterns $\mathbb{P} = \{(x^p, t^p)|p = 1, \ldots, |\mathbb{P}|\}$ is required. An input vector x^p corresponds to a say message $\mathcal{C} = (c_s, \ldots, c_0)$ with $s < 10$ and literals c_i from the alphabet A of 94 printable ASCII-128 characters. So, the discrete set of transmittable messages is $\mathcal{A} = \cup_{i=0}^{9} A^i$ which in our setting boils down to $|\mathcal{A}| \approx 5.1 \cdot 10^{19}$.

In regard to the target values t^p, however, we arrive at two sub-tasks relevant to the specific task of opponent pass announcement recognition:

(a) The problem of classifying whether an intercepted say message $\mathcal{C}$ contains a pass announcement or not. Accordingly, it holds $t^p \in \{true, false\}$.
(b) Given that the classifier from (a) states that $\mathcal{C}$ contains a pass announcement, the next logical challenge is to extract details of the pass announcement

from $\mathcal{C}$, such as its starting point or velocity. Thus, we obtain a classical regression problem with $t^p \in \mathbb{R}^l$ ($l = 4$ in case of pass announcements, as x and y components of the start position and velocity characterize the pass).

If we proceed on the assumption that the opponent team under consideration does announce its pass (this is a valid assumption for the 2D league), the training data set can be compiled easily by running a large batch of matches against the considered opponent team, recording its communication as well as observing its actual passes played. If, in so doing, a pass is accompanied by a say message $\mathcal{C}$ sent simultaneously to or shortly before the pass is played, then it is likely that $\mathcal{C}$ contains a pass announcement plus details of the intended pass.

3.2 Bit-Level Representation of Communicated Messages

As pointed out, the goal is to build and train a deep neural network into which some representation of the say message $\mathcal{C}$ is fed and whose output neuron(s) provide(s) decoded pass-related information. Intuitively, it seems tempting to feed a numeric representation of each letter c_i (e.g. its ASCII code) into the first layer of the network. Such an approach might indeed be expected to yield good results, if the payload to be transmitted is generally not distributed across multiple chars and if certain pieces of information were known to be located at a fixed position within $\mathcal{C}$. Given the limited communication bandwidth, however, these assumptions are unrealistic to be made. Accordingly, in [4] it has been suggested to employ a bit level representation of $\mathcal{C}$. Among other merits, such a representation will contain bit patterns that hint to the type of data encoded in the message as well as patterns that can be decoded to pass parameters[1].

Most importantly, a bit representation allows for the utilization of convolutional neural networks [6, 5] that perform one-dimensional convolution on the bit sequence in order to detect features that allow for classifying a message as containing a pass announcement or for extracting pass parameters. Therefore, any say message $\mathcal{C} \in \mathcal{A}$ is mapped to a bit sequence $b(\mathcal{C})$ using a function $b : \mathcal{A} \to \{0, 1\}^B$ where B is determined by the length of the message and the size of the underlying alphabet (in our case $B = 10\lceil \log_2 94 \rceil = 70$).

The authors of [4] suggest different ways of defining that function b, discussing in detail the motivation, advantages and limits of each suggested bit level encoding. In the rest of this paper, we stick to the "Base-$|A|$ Bit Level Representation" ($b_{|A|}$) which, according to [4], makes some assumptions on how opponent teams might have encoded their say messages, and which has brought about superior empirical results when using it as the basis of the decoding approach.

3.3 Model Architecture and Performance

We utilize the same deep convolutional neural network architecture as the one described in [4] (two convolutional layers, ReLu activations, max pooling, fully

[1] Under the assumption, that the opponent team does not employ some form of sophisticated data encryption or compression techniques before broadcasting messages. Hence, we proceed on the assumption that communicated data is not encrypted.

connected final layers, dropout, Adam optimizer) with one minor exception: For the task of extracting the numeric pass details (x/y of pass start position and velocity) we do not employ a single trained model with an output vector of length four, but four separately trained models with the same base architecture, but a single output neuron each (one of these for each of the four pass announcing variables, p_x, p_y, v_x, v_y). With four models using a single output neuron each and trained separately, we were able to achieve better generalization capabilities, i.e. the average test errors of the to-be-predicted four numeric pass details were significantly smaller compared to the monolithic model with a four-dimensional output (even when the latter was allowed to be trained for a much longer time).

For the general empirical performance of the approach, we again refer to [4]. Most notably, it is possible to train a reliable pass announcement classifier with as little as 50 sample passes observed (accuracy of 98.6%). For 20k training examples, the accuracy increases to 99.2%. Inferring pass announcement details reliably requires substantially more training samples. However, with 20k passes in the training data set, the average error of the pass start position is less than one meter (on the 2D playing field with 105m length and 68m width), and the average pass velocity error is generally less than $0.1\frac{m}{step}$ (where $v_{x/y} \in [-3\frac{m}{step}, 3\frac{m}{step}]$).

Based on these definitions, implementations and the reported decoding accuracies, our idea was to incorporate this approach to eavesdrop and understand opponent team communication into our competition team. This raised two questions. First, what engineering effort is required to make such an approach practically usable in an existing soccer simulation team. And second, what are the benefits in terms of possibly increased playing performance, if we succeed in enabling our team to reliably decode and exploit opponent team communication.

4 Implementation within the FRA-UNIted Framework

The learning approach outlined above had first been implemented in a prototypical manner utilizing TensorFlow's well-documented Python API. However, deploying this approach within the FRA-UNIted competition team, which is implemented in C++ entirely, issued quite an engineering challenge.

- Doing classification/regression with the trained networks utilized via Python scripts and using inter-process communication with our C++-based agent binary would have been a first option. But the resulting IPC overhead in conjunction with the need to possibly set up TF on competition machines render this approach impractical from our point of view.
- Porting the entire approach to C++ did not represent a valid option, too, since TF's C++ API does not enable access to optimizer classes such as the Adam optimizer for training the decoding models.
- We thus had to opt for a hybrid approach where the network topology definition and the training process of the deep networks takes place in the Python world. During matches, stored networks (TF checkpoints) are loaded via TF's C++ API and are utilized by our agent via a shared TF library that we built and that is dynamically loaded by the FRA-UNIted agent.

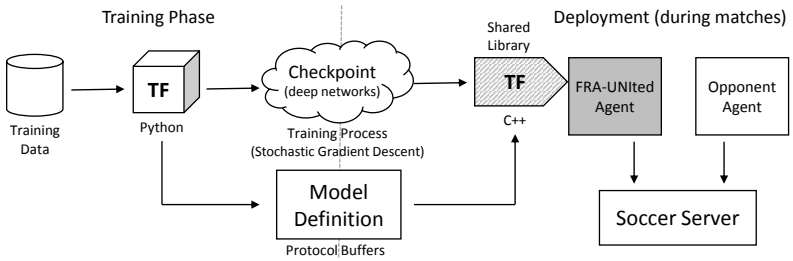

Fig. 2. Python scripts define the topology of the deep networks and use previously gathered data for training. Training results and model definitions are stored in separate files. The latter are generic for all teams, while the former are different for each opponent team we face. Via TF's C++ API, both files are then utilized by our agent.

The specific challenges and hurdles of creating and utilizing a shared C++ library that contains vast parts of TensorFlow as well as the technical and engineering details of our corresponding implementation can be found in our current team description paper [3].

5 Empirical Evaluation

In [15], the authors have presented the results of a study to measure the efficiency of inter-agent communication and its influence on the general playing performance of robotic soccer simulation teams. While in that paper the focus has been on the analysis of the structural and functional connectivity of the graph of communicating players, our main interest is on quantifying the differences in playing strength of a team that applies varying communication behaviors which is related to the notion of design points used by the authors of [15].

5.1 Communication Behavior Variants (CBV)

In all our experiments, we abbreviate our team FRA-UNIted as *Team A*. The version with which *Team A* participated in last year's world championships (RoboCup 2017) represents our baseline, labeled *A_Baseline17*. We selected the current world champion (Helios [2]) as the opponent (calling it *Team B*) against which to compare. More specifically, we selected both, its 2017 champion binary as well as its predecessor from RoboCup 2016 (labeled *B_Baseline17* and *B_Predecessor16*, respectively). In order to reliably quantify the contribution of communication to the overall performance of *Team A*, we adapted its baseline version, thus yielding the four communication behavior variants (CBVs) considered subsequently.

(a) *A_Baseline17*: Version of *Team A* used at the RoboCup 2017 tournament.
(b) *A_NoComm*: While no change to the agents' playing behavior has been made, the agents of *Team A* no longer use *any* form of inter-agent communication. So, this CBV should perform worse than the baseline.

(c) *A_NoOwnPasses*: Again, there is no change in the agents' playing behavior, but passes that are intended and/or are being played will no longer be communicated among the players of *Team A*. All other aspects of team-internal communication (like communicated player information or ball data) remain untouched.
(d) *A_OppPassExploit*: This communication behavior variant is in the center of our interest since it represents the baseline version enhanced by the implementation of the approach to eavesdrop and understand opponent pass announcements, whose basic ideas are described in Section 3.

It is important to stress that the actual *exploitation* of an intercepted opponent pass announcement in (d) has intentionally been implemented in a straightforward manner in order to facilitate an utmost fair assessment of the impact of the approach. Each decoded opponent pass announcement is treated in exactly the same manner as a pass announcement received from a teammate (like in (a) or (b)). Accordingly, no additional logic or special cases for opponent passes were programmed, which allows us to assess the benefits of having wiretapped the opponent team as accurately as possible. Hence, from the point of view of an *A_OppPassExploit* agent, there are just "a few more" pass announcements compared to the baseline version of the agents. As a consequence, the advantage of (d) over (a) is, essentially, time: An *A_OppPassExploit* agent will, in general, know earlier about an opponent pass than a baseline agent and, thus, will be able to react on this faster. Besides the changes mentioned, the only necessary modification of *A_OppPassExploit* agents compared to their baseline counterparts concerned their listening attention-to behavior: In order to be able to reliably receive say messages from the opponent ball holder h, the listening attention had to be put to h instead of putting it to some teammate (as a baseline agent would do, by contrast), if h controls the ball.

5.2 Distribution of Communication Data

In order to assess the communication restrictions imposed on the *A_NoComm* and *A_NoOwnPasses* CBVs, it is advisable to quantify the amount of communication data that is on average received by a *A_Baseline17* agent. We refer to the publicly available source code release of *Team A* which reveals the inner workings of its communication policy [11]. Table 1 shows that 70% of all communicated pieces of information are player-related, i.e. positions of teammates or opponents recently seen. By contrast, pass-related information make up for only 0.8% of all communicated data chunks, corresponding to 96.3 pass announcements and 2.6 pass requests per player per game on average. Moreover, that table summarizes the average distribution of the total amount of data sent among the five considered types of soccer-related information (in total number of bits received and the corresponding share of communication channel usage per type). Essentially, only 1.7% of the overall communication bandwidth is used for announcing passes to teammates.

As a consequence, *A_NoOwnPasses* agents (which do not receive pass announcements from teammates) disregard circa 0.8% of all data chunks that are

	Overhead	Ball-Rel	Pass-Rel	Player-Rel	Strategy-Rel
Avg. number of payload units	714.4	615.5	98.9	8535.8	3030.4
sent per match (total and share)		5.0%	0.8%	69.5%	24.7%
Avg. amount of bits sent per match	3571.9	104.4k	3102.8	153.6k	15151.9
Share of comm. channel usage	1.9%	5.6%	1.7%	82.6%	8.2%

Table 1. Utilization of the Communication Channel by *A_Baseline17*: Besides overhead (headers etc.), 4 soccer-related categories of information are shared across agents.

communicated in our team, which makes up for 1.7% of the overall data payload. By contrast, *A_Baseline17* disregards nothing, and *A_NoComm* disregards the entire communication.

5.3 Empirical Methodology

All four communication behavior variants of *Team A* were matched against both versions of *Team B*. For each combination, 5000 matches were played in order to form score averages and, hence, to account for the stochastic nature of the Soccer Server. To assess the overall team playing strength and to analyze the impact of altered communication schemes, we adopt the perspective of *Team A* and, for the rest of this paper, focus on the following performance measures.

- μ_a: average number of goals scored during a single match by *Team A* with belonging standard deviation σ_a, calculated over the set of matches played.
- μ_b: average number of goals scored by *Team B*, i.e. the average number of goals conceded by *Team A* with belonging standard deviation σ_b.
- μ_p: expected number of points *Team A* gains from a match against *Team B* on average, when a victory is rewarded with three points, a draw with one, and a defeat with none.

Given that the modifications to the communication behavior of the CBVs will most likely have affected the standard deviations of the performance measures considered, we applied an unequal variance t-test (also known as Welch test [14]) in order to determine whether any empirically observable change of μ_a or μ_b is statistically significant (and if so, at which confidence level).

5.4 Results

Figure 3 shows the variability in playing strength of the four *Team A* CBVs considered. Here, *A_Baseline17* is considered as the baseline (100%) against which the three other variants are compared with respect to the relative amount of goals scored and conceded against both versions of *Team B*. Consistently, an increased utilization and exploitation of communicated information brings about an increased overall performance. Interestingly, when playing against *B_Baseline17*, the activation of our wiretapping and opponent pass announcement decoding approach yields an increase in the number of goals shot by 8.8% and a simultaneous decline in the number of goals conceded by 4.4%.

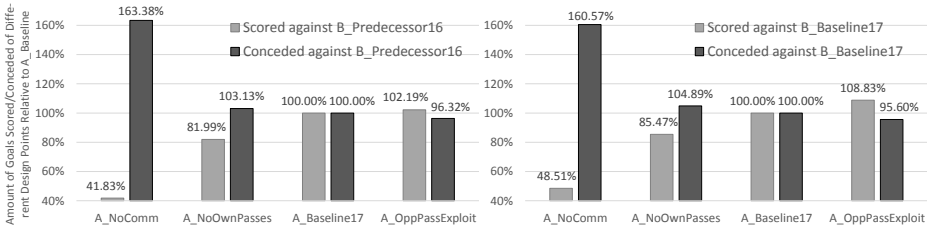

Fig. 3. Taking CBV *A_Baseline17* as the ground truth, these charts visualize the relative changes in μ_a and μ_b when the other three considered CBVs are matched against *Team B* (left: against *B_Predecessor16*, right: against *B_Baseline17*).

While Figure 3 shows performance measures relative to *Team A*'s baseline, the left chart in Figure 4 highlights absolute values of μ_a and μ_b for all combinations of CBVs of *Team A* having played against both *Team B* versions. The lengths of the line segments in both data series convey a good impression of how the overall playing capabilities of *Team A* are impacted by switching off/on the entire team-internal communication, just team-internal pass announcements, and eavesdropping and exploiting opponent pass-related communication. Apparently, the impact and usefulness of the opponent pass decoding, is more distinct when playing against *Team B*'s 2017 champion version than when playing against its 2016 predecessor, though the former has, of course, a generally higher playing strength than the latter. It is also worth noting that, when testing against *B_Baseline17*, the gain/loss of switching on/off the exploitation of *Team B*'s decoded pass announcements is almost as pronounced as the one resulting from switching on/off our team-internal pass announcements.

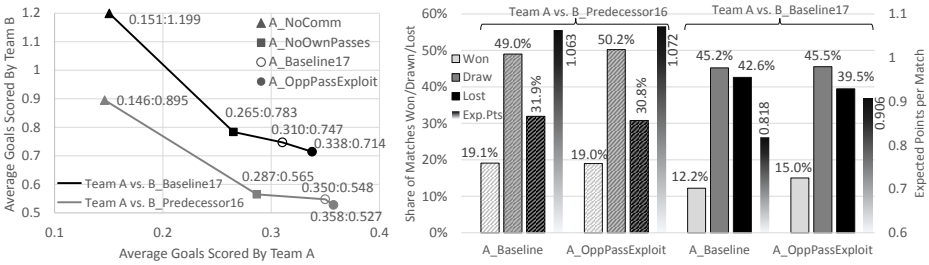

Fig. 4. Left: Average scores of the four *Team A* CBVs when facing both *Team B* versions considered. Right: Share of matches won/drawn/lost as well as expected points μ_p when *A_Baseline17* and *A_OppPassExploit* face both *Team B* versions.

The right part of Figure 4 concentrates on the expected points μ_p to be obtained when playing against the two versions of *Team B*, but just focuses on a comparison of the two CBVs *A_Baseline17* and *A_OppPassExploit*. To do so, it visualizes the share of matches won/drawn/lost by *Team A* as well as the exact

value of μ_p (secondary axis). With respect to opponent *B_Predecessor16*, the exploitation of heard opponent pass announcements reduces the percentage of games lost by roughly 1%, increasing the share of draws accordingly, thus resulting in an improvement of μ_p by about 1%, too. By contrast, when evaluating against the current world champion *B_Baseline*17, we observe an increase of $\approx 3\%$ in the share of matches won at a constant level of draws, which amounts to a 10.8% growth of the expected points per match (from 0.818 to 0.906).

Opponent	Measure	CBV with conf. level of a significant change vs. *A_Baseline17*							
		A_NoComm	$p_{a/b}$	*A_NoOwnPass*	$p_{a/b}$	*A_Baseline17*	*A_OppPassExp*	$p_{a/b}$
B_Predecessor16	$\mu_a \pm \sigma_a$	$0.146 \pm .490$	.001	$0.287 \pm .536$	.001	$0.350 \pm .578$	$0.358 \pm .575$	.25
	$\mu_b \pm \sigma_b$	$0.895 \pm .966$	.001	$0.565 \pm .730$	.1	$0.548 \pm .746$	$0.527 \pm .739$	.1
B_Baseline17	$\mu_a \pm \sigma_a$	$0.151 \pm .380$	.001	$0.265 \pm .506$	.001	$0.310 \pm .553$	$0.338 \pm .555$	.01
	$\mu_B \pm \sigma_b$	1.199 ± 1.135	.001	$0.783 \pm .895$	.01	$0.747 \pm .878$	$0.714 \pm .842$	.025

Table 2. Performance measures μ_a and μ_b for all CBV pairings. p_a and p_b stand for the error levels at which a change in μ_a and μ_b (i.e. a change from $\mu_{a/b}^{A\_Baseline17}$ to any other $\mu_{a/b}$ value) is statistically significant.

Table 2 summarizes the values of performance measures μ_a and μ_b for $n = 5000$ game repetitions for each CBV playing against both *Team B* versions. Also, we report the significance levels at which the null hypothesis for the Welch test has to be rejected (null hypothesis: performance measure did not improve/worsen (compared to *A_Baseline17*) due to switching on/off the communication feature of the respective CBV). While the test statistic allows us to confirm the expected changes at very low error rates in most cases, for the evaluation of *A_OppPassExploit* versus the older 2016 version of *Team B* we can attest the expected improvements in μ_a and μ_b at an error level of 0.1 and 0.25, only.

5.5 Discussion

In the experimental evaluation at hand, our selected opponent (*Team B*) was the current world champion. Thus, all conclusions refer to this opponent in the first place. However, as pointed out by [4], similar or even better communication learning performance can be expected, when playing against other 2D top teams. Besides, by having selected the currently best team in the world, we have set a high standard and it is standing to reason that our findings can be generalized to (at least several) weaker teams. Clearly, when switching to another opponent a separate decoding model in the form of a deep convolutional neural network would have to be trained beforehand. Also, we should emphasize the fact that the advantages reported can only be exploited in real tournament games under the assumption that the opponent does not change or encrypt its communication.

The disk space requirements of *Team A* increase substantially, when incorporating the presented approach into our team. Having consumed $6.9MB$ in total in its RoboCup 2017 version (*A_Baseline17*), the new working binary

(*A_OppPassExploit*) now requires $110MB$ where $96MB$ are consumed by the created TensorFlow shared library and $7.3MB$ are due to newly trained neural networks for communication decoding.

The computational burden caused by the TensorFlow-based opponent communication decoding is acceptable. On a 4-core i7 with 3GHz (with all 22 players plus 2 coaches plus the Soccer Server running in parallel on this single machine) *without* GPU support a say message classification requires between 3 and 4 milliseconds. The subsequent determination of pass start and velocity vectors costs between 9 and 10 milliseconds on average. Given that during competitions teams are allowed to employ several machines (typically only 3-4 agents play on a multi-core machine, i.e. a separate CPU core is available for each agent), the mentioned calculation times are likely to be around or even below 1.5 milliseconds. With respect to the required training times of the deep neural networks, which of course strongly depend on the hardware used and on the availability of powerful GPUs, we refer to the numbers we reported in [4].

It is worth noting that an agent of CBV *A_OppPassExploit* does receive, decode, and exploit 66.2 passes from *Team B* during one full match on average, and that this is almost 70% of the number of pass announcements they receive from their own teammates (96.3 on average per game). However, a substantial amount of these opponent pass announcements are "safe passes" which, despite the fact that we hear and understand them, by no means, allow for tackling the pass receiver or in conquering the ball, immediately. We found that an *A_OppPassExploit* agent gets to know that an opponent pass is being played on average 2.2 time steps earlier than an *A_Baseline17* agent which has to rely on its (restricted, noisy, and non-omnidirectional) vision system to see the pass. Given the comparatively small number of exploitable opponent passes and, hence, intuitively small qualitative influence on the overall course of the game, the reported quantitative impact on the general playing strength of *Team A* is remarkable.

6 Conclusion

It has been argued by numerous authors that the utilization of team-internal communication is highly beneficial in soccer simulation 2D. In this paper, we have substantiated that claim by two means. On the one hand, we have compared the empirical playing strength of our team (FRA-UNIted) when disabling certain paths of communication across teammates. On the other hand, we have utilized a deep learning-based approach for decoding the contents of say messages broadcast by the opponent team. In so doing, we could show that the playing performance of a team that wiretaps an opponent and exploits intercepted information (in our case pass announcements) can be boosted significantly.

Since the mentioned deep learning-related part of the approach relies on a TensorFlow-based implementation, a critical question concerns the practicability of our approach. Powerful machine learning libraries and their APIs evolve fast. In regard to the fact that our team binary should retain easily deployable on any machine (e.g. during competitions or for reproducibility) it was our goal to utilize

the required TensorFlow functionality into our team with as little installation or maintenance effort as possible. Hence, our delineations in Section 4 (and, in addition, in our current team description paper [3]) are meant as an aid for teams in the simulation league and beyond which intend to merge their team sources with TensorFlow utilizing its C++ API.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015), http://tensorflow.org/, software available from tensorflow.org
2. Akiyama, H., Nakashima, T., Fukushima, T.: HELIOS2017: Team Description (2017), www.robocup2017.org/file/symposium/soccer_sim_2D/TDP_HELIOS 2017.pdf, Supplementary to RoboCup 2017: Robot Soccer World Cup XXI
3. Gabel, T., Klöppner, P., Godehardt, E.: FRA-UNIted – Team Description 2018 (2018), tgabel.de/cms/fileadmin/user_upload/documents/Gabel_EtAl_FU-18.pdf, Supplementary material to RoboCup 2018: Robot Soccer World Cup XXII
4. Gabel, T., Tharwat, A., Godehardt, E.: Eavesdropping Opponent Agent Communication Using Deep Learning. In: Proceedings of Multi-Agent System Technologies (MATES 2017). pp. 205–222. Springer, Leipzig (2017)
5. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2017)
6. LeCun, Y.: Generalization and Network. Design Strategies. Technical Report CRG-TR-89-4, University of Toronto (1989)
7. MacAlpine, P.: Multilayered Skill Learning and Movement Coordination for Autonomous Robotic Agents. Ph.D. thesis, University of Texas at Austin, USA (2017)
8. Mota, L., Reis, L.: An Elementary Communication Framework for Open Cooperative RoboCup Soccer Teams. In: Proceedings of the 3rd International Workshop on Multi-Agent Robotic Systems. pp. 97–101. Scitepress, France (2007)
9. Noda, I.: Soccer Server: A Simulator of RoboCup. In: Proceedings of the AI Symposium 1995. pp. 29–34. Japanese Society for Artificial Intelligence (1995)
10. Prokopenko, M., Wang, P.: Evaluating Team Performance at the Edge of Chaos. In: D. Polani, B. Browning, A. Bonarini, K. Yoshida, editors, RoboCup 2003: Robot Soccer World Cup VII, LNCS. pp. 89–101. Springer, Padua, Italy (2003)
11. Riedmiller, M., Gabel, T., Schulz, H.: Brainstormers 2D: Public Source Code Release 2005. Technical Report, University of Osnabrück (2005), sourceforge.net/projects/bsrelease/files/bs05publicrelease.documentation.pdf
12. Stone, P., Veloso, M.: Communication in Domains with Unreliable, Single-Channel, Low-Bandwidth Communication. In: Drogoul, A., Tambe, M., Fukuda, T. (eds.) Collective Robotics, pp. 85–97. Springer, Berlin, Germany (1998)
13. Stone, P., Veloso, M.: Multiagent Systems: A Survey from a Machine Learning Perspective. Autonomous Robots 8(3), 345–383 (2000)
14. Welch, B.: The Significance of the Difference Between Two Means When the Population Variances Are Unequal. Biometrika (29), 350–362 (1938)
15. Zuparic, M., Jauregui, V., Prokopenko, M., Yue, Y.: Quantifying the Impact of Communication on Performance in Multi-Agent Teams. Artificial Life and Robotics 22(3), 357–373 (2017)