Improving Prediction of User Cognitive Abilities and Performance for User-Adaptive Narrative Visualizations by Leveraging Eye-Tracking Data from Multiple User Studies

by

Alireza Iranpour

B.Sc., Sharif University of Technology, 2018

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

October 2021

The following individuals certify that they have read, and recommend to the Faculty of Graduate and Postdoctoral Studies for acceptance, the thesis entitled:

Improving Prediction of User Cognitive Abilities and Performance for User-Adaptive Narrative Visualizations by Leveraging Eye-Tracking Data from Multiple User Studies

submitted by Alireza Iranpour in partial fulfillment of the requirements for

the degree of Master of Science

in Computer Science

Examining Committee:

Cristina Conati, Computer Science, UBC
Supervisor

David Poole, Computer Science, UBC
Supervisory Committee Member

Abstract

Previous work leveraged eye-tracking to predict a user's levels of cognitive abilities and performance while reading magazine style narrative visualizations (MSNV), a common type of multimodal document which combines text and visualization to narrate a story. The eye-tracking data, used for training the classifiers, came from a user study, called control, where subjects simply read through MSNVs without receiving any type of adaptive guidance, otherwise known as the control condition. The goal was to capture the relationship between users' normal MSNV processing and their levels of cognitive abilities and performance and use that to drive personalization. In addition to the control study, two other user studies were also previously conducted to investigate the benefits of adaptive support, also known as adaptive studies. In these studies, subjects were provided with gaze-based interventions to facilitate their processing of the MSNVs.

In the control study, there was no intervention, and the MSNVs did not adapt to the users in any way because the idea was to make the predictions based on users normal, unguided MSNV processing and then use those predictions to deliver the appropriate adaptations. In the adaptive studies, however, interventions influenced the way subjects processed the MSNVs. As a result, their gaze behavior did not represent how they would have behaved in the intended control condition, and a classifier merely trained on the eye-tracking data from these studies would not learn the proper relationship.

In this thesis, we propose different strategies for combining the additional eye-tracking data from the adaptive studies with our original data from the control study to mitigate the potential differences and to form more consistent combinations conducive to improved performance. Our results show that the additional eye-tracking data can significantly improve the accuracy of our classifiers.

Lay Summary

Eye-tracking has been utilized for predicting users' long-term characteristics as well as their transient states in order to drive personalization. Processing information visualization, due to its perceptual nature, is another task where eye-tracking has proven to be useful for inferring user characteristics to support AI-driven personalization. Particularly, in the context of processing magazine style narrative visualizations (MSNV), a common type of multimodal documents which combines text and visualization, eye-tracking has been used to predict users' cognitive abilities and performance. This work strives to improve the accuracy of these predictions by taking advantage of additional eye-tracking data which could potentially help model users' behavior more precisely for more accurate predictions.

Preface

This thesis is based on multiple research projects conducted in the Human-Centered Artificial Intelligence Laboratory at the University of British Columbia and advised by Prof. Cristina Conati.

Specifically, this thesis extends the work conducted in the following paper:

- Oswald Barral, Sébastien Lallé, Grigorii Guz, Alireza Iranpour, and Cristina Conati. 2020. Eye-Tracking to Predict User Cognitive Abilities and Performance for User-Adaptive Narrative Visualizations. In Proceedings of the 2020 International Conference on Multimodal Interaction (ICMI '20).

(As one of the authors of the above work, I performed the optimization of model hyper-parameters and performance evaluation using a nested cross-validation)

And uses the data previously collected in the following works:

- Dereck Toker, Cristina Conati, and Giuseppe Carenini. 2019. Gaze analysis of user characteristics in magazine style narrative visualizations. User Model. User-Adapt. Interact. to appear, (2019)
- S. Lallé, D. Toker and C. Conati, Gaze-Driven Adaptive Interventions for Magazine-Style Narrative Visualizations, in IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 6, pp. 2941-2952, 1 June 2021
- S. Lallé, T. Wu and C. Conati, Gaze-Driven Links for Magazine Style Narrative Visualizations, 2020 IEEE Visualization Conference (VIS), 2020

Figures 1, 2, 3, 4, and 11 are used with permission from applicable sources. Portions of the introductory text are modified from introductory material from Barral et al. (2020) of which I am an author. Chapters 4.2 and 4.3 are used with permission from Barral et al. (2020). I performed all additional experiments.

The fieldworks reported in Chapter 3 were all approved by the University of British Columbia's Research Ethics Board [certificate #H04-80596].

Table of Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank the University of British Columbia's Computer Science Department for allowing me to research my master's thesis and providing me with a state-of-the-art education in Computer Science. Particularly, I would like to thank my supervisor Dr. Cristina Conati for providing me with the help and guidance I needed to conduct this research.

Special thanks are owed to my parents, whose have supported me throughout my years of education, both morally and financially.

Dedication

I would like to dedicate this work to my parents for their love and support through all of the ups and downs of my studies. Thank you for always believing in me and supporting me no matter what. You are the strongest pillars that held me up during this process, and this work would not have been possible without you. I love you.

1. Introduction

Eye-tracking has been utilized for predicting users' long-term characteristics as well as their transient states in order to drive personalization. Examples include relevance feedback for information retrieval [5], cognitive load assessment for emergency situations [6], learning gains with educational software [7], and automatic detection of personality traits [8]. Processing Information Visualization (InfoVis), due to its perceptual nature, is another task where eye-tracking has proven to be useful for inferring user characteristics to support AI-driven personalization. Particularly, in the context of processing magazine style narrative visualizations (MSNV), a common type of multimodal documents which combines text and visualization (e.g., Figure 1), eye-tracking has been used to make real-time predictions for three user cognitive abilities, namely reading proficiency, visualization literacy, and verbal working memory, all relevant to decide what to adapt to, and two measures of task performance, namely task comprehension and task completion time, relevant to decide the triggering of adaptations (i.e., when to adapt) [1].

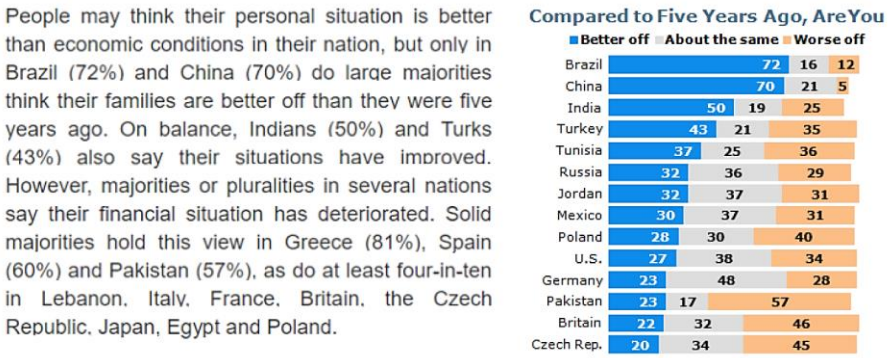

Figure 1: Sample MSNV

MSNVs convey information in two different modalities. As a result, processing such documents can be challenging due the need to split attention among two sources of information, a phenomenon known as the split-attention effect which can increase cognitive load [9] and cause difficulties for users with lower levels of cognitive abilities. Hence, accurate prediction of user states and characteristics is necessary for an effective personalization where the system delivers the right adaptation to the users who need help without distracting or overwhelming those who do not require these types of support.

Previous work [1], leveraged eye-tracking to predict a user's levels of cognitive abilities and performance while reading MSNVs. The eye-tracking data, used for training the classifiers, came from a user study where subjects simply read through MSNVs without receiving any type of adaptive guidance, otherwise known as the control condition (control study). The goal was to capture the relationship between users' normal MSNV processing and their levels of cognitive abilities and performance and use that to drive subsequent personalization.

Nevertheless, in addition to the control study, two other user studies were also conducted to investigate the benefits of adaptive support (adaptive studies). In these studies, subjects were provided with gaze-based interventions to facilitate their processing of the MSNVs. Specifically, one study [2], used highlighting intervention to dynamically emphasize relevant data points in the MSNV chart (bars) as users read through their corresponding references in the narrative text. The other study [3], augmented

the bar highlighting interventions from [2] by underlining the reference sentence of the currently highlighted bars and connecting them with a set of lines (links).

The interventions delivered in the adaptive studies influenced the way subjects processed the MSNVs. As a result, their normal gaze behavior was affected and rendered inconsistent with that of the control group. For that reason, these eye-tracking data might not help our classifiers learn the desired relationship between users' normal unguided gaze behavior and their levels of cognitive abilities and performance more effectively.

In this thesis, we propose different strategies for combining the additional eye-tracking data with our original data, intended to mitigate the potential differences and to form a more consistent combination conducive to improved performance. Our results show that our winning strategy can increase the efficacy of our combined data and significantly improve accuracy for most of our target measures. In particular, we show that for VerbalWM, our combined data achieve a significantly higher accuracy of 0.64 than the baseline of 0.51 which was previously the best performance possible for this cognitive ability within-task. We also show that for ReadP and Task Time, our combined data enable more accurate predictions earlier allowing for timelier adaptations. Specifically, we achieve peak accuracies of 0.75 and 0.73, respectively, which are significantly higher than their previous peak of 0.68. For VisLit too, we show that an early prediction can be significantly improved from 0.58 to 0.64 with the combined data, thus demonstrating the effectiveness of the additional eye-tracking data in improving our prediction performance.

2. Related Work

2.1. Eye-Tracking for User Modelling and Adaptation

Eye-tracking for gaze driven adaptation (i.e., adaptation which reacts to specific user gaze patterns) has been investigated in several domains (see [4] for an overview). In educational settings, D'Mello et al. [10] used gaze-sensitive prompts to promote student engagement by reorienting their attention when they looked away from the screen. Alt et al. [11] customized web contents based on what information users looked at while browsing e-commerce web pages. In information visualization, gaze-based adaptation was used to support map reading by dynamically adapting both the placement and the content of the legend [12] or de-emphasizing parts of the map which were not within the user's focus of attention [13]. Lallé et al. provided dynamic gaze driven interventions in the context of narrative visualizations by highlighting relevant data points in the visualization [14] and displaying links between sentences and their corresponding data points as users read through the text component [3]. Research has shown that eye-tracking can reveal more about users than simply where they look. In particular, eye-tracking has been used for user modeling in real-time by predicting a user's task performance, as well as their short-term states and long-term characteristics which can drive adaptive support.

Predicting task performance

Eye-tracking has been utilized for predicting different measures of task performance useful for deciding when to adapt (i.e., when the user is predicted to have a low performance on the task). Examples include predicting a user's task speed in performing visual search tasks [15] and processing bar and radar charts [16]. Prediction of task speed alone, however, is not adequate for driving adaptations as there is usually a trade-off between speed and accuracy. For instance, a user may be slower but more accurate. Other work predicted task comprehension when reading plain texts with no visualizations using a combination of gaze and speech data [17]. Barral et al. successfully predicted task time, as well as task comprehension, from eye-tracking data in multimodal documents that combine text and visualizations [1]. There also exists work on predicting domain specific measures of performance such as learning gains with tutoring systems [18][19] and problem-solving performance in puzzle games [20].

Predicting user states and characteristics

Eye tracking has also been used for predicting various short-term affective and cognitive states. Examples include predicting emotions [21][22] and mind wandering [23][24] while performing educational tasks and watching videos, intention in multiplayer games [25], interest when viewing web documents [26], and mental workload while performing document editing and rout planning tasks [27]. In information visualization (InfoVis), user confusion [28], interest [29], and familiarity with visualizations [30] have also been predicted using eye-tracking.

In addition to short-term states, long-term cognitive abilities and personality traits have also been predicted from eye-tracking. These characteristics can determine the type of adaptation. In particular, previous work predicted cognitive abilities, which influence visual processing, such as perceptual speed, verbal working memory (Verbal WM from now on), visual working memory, and visual scanning [31][32][33]. These cognitive abilities were predicted while users processed simple bar and radar charts [32][34], as well as a decision-making interface featuring a map and a deviation chart [31]. Eye-tracking

has also been used to predict reading ability when reading text [35], an ability which influences user comprehension [36]. Other works predicted personality traits while watching videos [37]. Song et al. [38] inferred user preferences for improving recommendations. Barral et al. [1] used eye-tracking to predict two long-term cognitive abilities shown to influence the processing of MSNVs [39], namely visualization literacy (i.e., the ability to use common data visualizations in an efficient and confident manner [40]), and reading proficiency (i.e., the vocabulary size and reading comprehension ability in English [41]), as well as verbal working memory (i.e., amount of verbal information (e.g., words) that can be temporarily maintained and manipulated in the working memory [42]).

2.2. Eye-tracking in Multimodal Documents

There has been little work on eye-tracking for user modeling and adaptation in the context of multimodal documents. Previous work used instructional material which contained text and pictures (not visualizations) [21][43][44]. Specifically, these works used eye-tracking for predicting students' learning outcome [44], emotions [21], and motivational goals [43]. Other works used eye-tracking for evaluation of how users process multimodal webpages containing text and pictures [45][46].

Despite the commonness of multimodal documents with embedded visualizations, such as narrative visualizations, only a few works have studied gaze-driven adaptation in this context. Lallé et al. used users' reading patterns, as captured by an eye-tracker, to deliver highlighting [2], as well as link [3] interventions in MSNV documents. They found that while highlighting interventions can improve comprehension of MSNV documents in users with lower levels of visual literacy, users with higher levels of this cognitive ability experienced a drop in comprehension [2]. Link interventions, on the other hand, improved comprehension in both groups [3].

Toker et al. [39] further studied the impact of cognitive abilities on task performance and gaze behaviors in non-adaptive MSNVs. They found that verbal WM affects overall reading time and processing of the text, with low verbal WM users taking significantly more time and looking significantly longer at the MSNV text than their counterparts. They also found that reading proficiency affects task speed and processing of the key components of the MSNV visualization, with low reading proficiency users being significantly slower due to more transitions to and from the labels and data points in the visualization. These findings show that users may benefit from adaptive support tailored to their levels of cognitive abilities and expected performance when reading MSNVs. Barral et al. [1] provided a successful first attempt at predicting these cognitive abilities and performance measures in MSNVs to drive adaptive support. In this work, we will improve the accuracy of these predictions by leveraging the additional eye-tracking data from the adaptive MSNV studies [2][3] which used the exact same documents.

3. Datasets

The datasets, used in this thesis, were generated in three different user studies from previous works. One study with non-adaptive MSNVs, also known as control [39], and two others with adaptive MSNVs (also known as bar [2], and link [3]). The data collection procedure was similar for all three studies.

Table 1: Summary statistics of study groups

	Ctrl	**Bar**	**Link**
Size	56	50	30
Age	Mean= 28, SD= 11 Range= (19-69)	Mean= 26, SD= 8 Range= (18-59)	Mean= 27, SD= 10 Range= (18-57)
Gender	57% female	60% female	57% female

During each of the user studies, the raw gaze data of the participants were captured using a Tobii T-120 eye-tracker with a sampling rate of 120 Hz on a screen display of 1280 × 1024 pixels. At the beginning of the session, the tracker was calibrated for each participant and the baseline pupil size was collected. Subjects were then given the task of reading an MSNV document on the screen and would signal, upon completion, by clicking "next" at the bottom of the MSNV. After that, they were presented with a set of questions intended to elicit their opinion of the document and to evaluate their comprehension of the relevant concepts discussed in it. Participants repeated this task with different MSNVs (15 times in the control study and 14 times in the adaptive ones) in a fully randomized order with no time constraint. Each MSNV consisted of a self-contained excerpt of a real-world document and one accompanying bar chart (see Figure 1). Next, each participant took a set of standard psychological tests to assess their cognitive abilities (long-term characteristics which would not vary over the course of the study). These tests included the Bar Chart Test for visualization literacy [40], the X_Lex Test for reading proficiency [41], and the OSPAN Test for verbal WM [50]. Table 1 reports the size and demographics of each study group.

The control study did not involve any type of intervention (Figure 2). The adaptive studies, on the other hand, provided users with adaptive guidance to facilitate processing of the MSNVs. Below we describe the type of intervention used in each adaptive study.

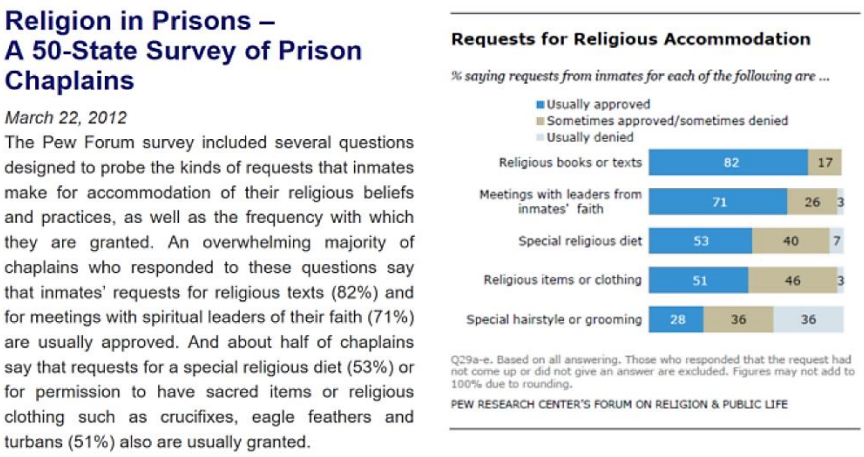

Figure 2: Sample MSNV with no intervention

Bar Highlighting Intervention

The highlighting intervention used in [2] was designed to dynamically emphasize relevant data points in the MSNV chart (bars) as users read through their corresponding references in the narrative text. The objective was to drive the user's attention to the appropriate data points in the visualization when it was most relevant (i.e., when the user was attending to the reference sentence in the text). Bars were highlighted by having their borders thickened in black. Figure 3 shows an example, where the two bars at the bottom of the chart are highlighted as the user reads the last sentence in the text that references them. There was no highlighting of the references in the text. Highlights were accumulated as users read through new references in the text with the previous ones having their outlines desaturated to help distinguish between those and the most recent ones (see top two bars in Figure 3).

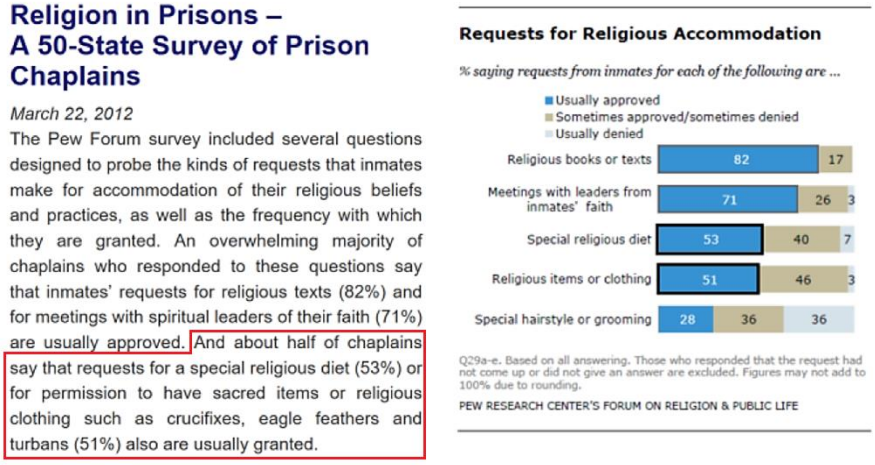

Figure 3: Sample MSNV with the bar intervention
(The red box indicates the paragraph to which the highlighted bars correspond. The box was not part of the intervention)

Link Intervention

The link intervention (Figure 4) used in [3] augmented the bar highlighting interventions from [2] with two additional components:

1. Underlining the reference sentence of the currently highlighted bars
2. A set of lines connecting the underlined reference to the highlighted bars (links)

The link intervention was designed to address some of the usability issues associated with the bar intervention as identified in [2]. In particular, underlining of the references was meant to address the reported difficulty of retrieving a sentence to resume reading after a switch of attention to the chart, and the links addressed the reported lack of noticeability of the highlighted bars.

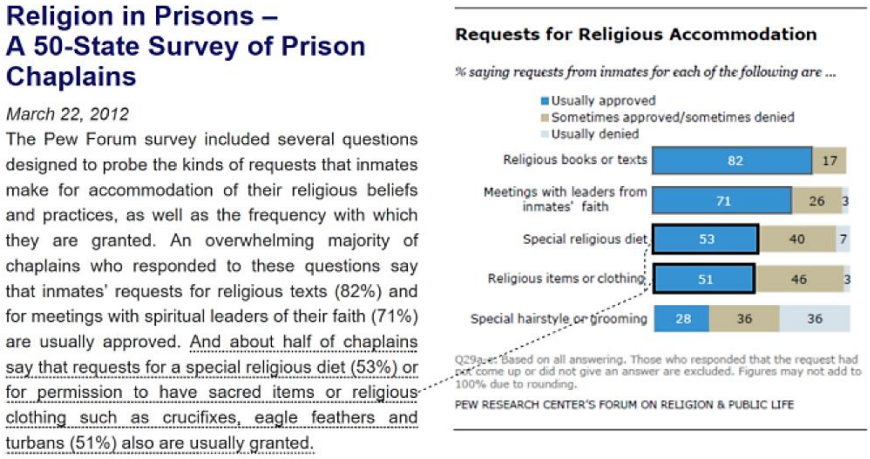

Figure 4: Sample MSNV with the link intervention

4. Classification Experiments

We leveraged users' eye-tracking data as they processed MSNV documents to build classifiers for predicting binary labels of three cognitive abilities, namely visualization literacy (VisLit), reading proficiency (ReadP), and verbal working memory (VerbalWM), which have been known to impact MSNV processing, as well as two measures of task performance, namely task completion time in seconds (Task Time), and task accuracy (proportion of comprehension questions answered correctly). The binary labels were generated by splitting participants into two classes of high and low. We are interested to find out whether the additional eye-tracking data from [2] and [3] can improve our classification accuracy in the control condition from [1]. For that reason, we will evaluate the performance of our classifiers only on the control data which represent users' gaze behavior in that condition.

4.1. Score Statistics

In this section, for each of the three cognitive abilities and the two task performance measures, we will provide and discuss summary statistics of the test scores used for splitting the participants. We followed a median split approach as used in [1]. In this approach, samples that fall above the median are assigned to the high class and those below to the low class. Samples that end up on the median are moved to either the high or the low class. For each measure, we will discuss how these samples were assigned.

Visualization Literacy (VisLit)

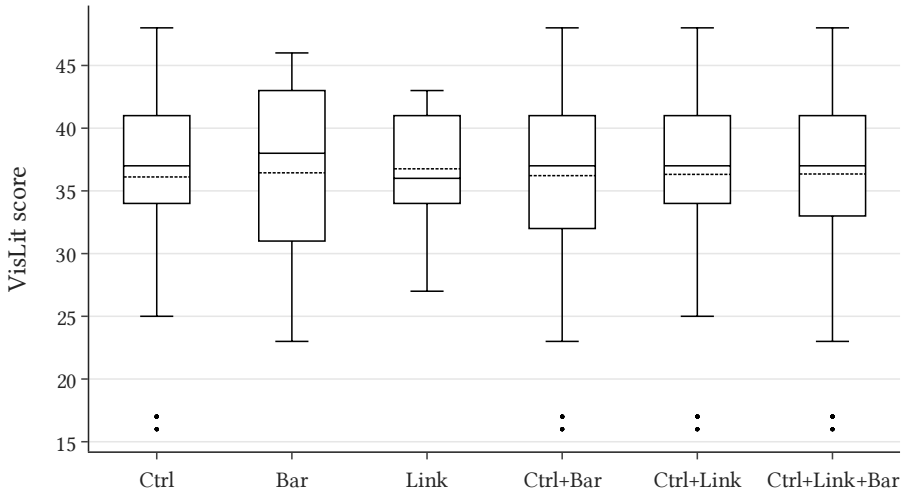

Figure 5: VisLit score box plots

Table 2: VisLit score stats

	Ctrl (54)	**Bar** (49)	**Link** (26)	**Ctrl+Bar** (103)	**Ctrl+Link** (80)	**Ctrl+Link+Bar** (129)
Mean (SD)	36.1 (6.7)	36.8 (5.8)	37.0 (4.6)	36.4 (6.3)	36.4 (6.1)	36.3 (6.2)
Median (M)	37	38	36	37	37	37
<M | = | >M	**23** | **6** | 25	**23** | **5** | 21	**12** | **2** | 12	**45** | **7** | 51	**37** | **6** | 37	**59** | **7** | 63

In terms of VisLit, the Ctrl dataset has 6 participants with the median score. In the previous work [1], these samples were assigned to the smaller group (i.e., the low class) as this would result in a more balanced distribution. Since we would be evaluating our models on the control data points, for consistency, we handled the median samples of all other datasets based on the median of the Ctrl. That is, for Bar, which has a larger median than Ctrl, we assigned the median samples to the high class, and for all other datasets, we assigned them to the low class.

Reading Proficiency (ReadP)

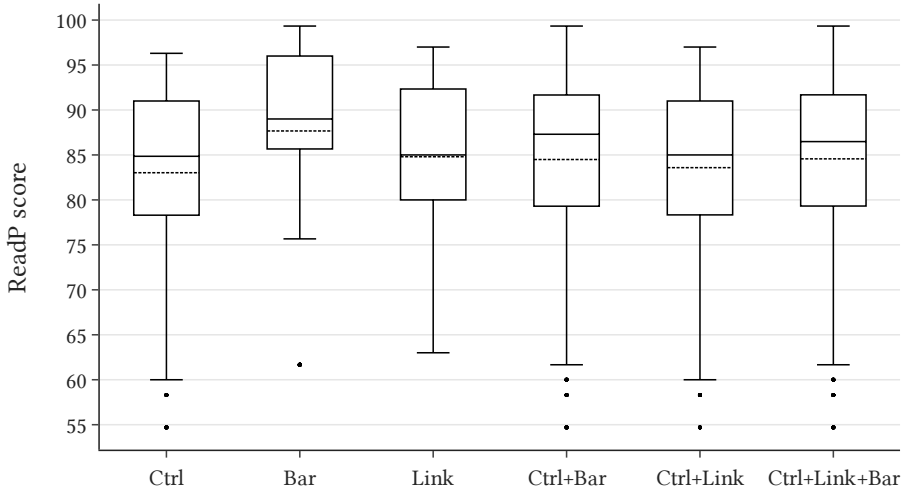

Figure 6: ReadP score box plots

Table 3: ReadP score stats

	Ctrl (54)	**Bar** (49)	**Link** (26)	**Ctrl+Bar** (103)	**Ctrl+Link** (80)	**Ctrl+Link+Bar** (129)
Mean (SD)	83 (10.2)	88.8 (9.3)	85.0 (8.4)	85.8 (10.2)	83.7 (9.6)	84.6 (9.8)
Median (M)	84.9	91	85	88.3	85	86.5
<M | = | >M	27 | 0 | 27	24 | **2** | **23**	12 | **2** | **12**	49 | **4** | **50**	39 | **3** | **38**	63 | **2** | **64**

In terms of ReadP, performing the median split on the Ctrl dataset resulted in a perfect class balance with no samples on the median. As a result, given that the median scores of all other datasets are greater than

that of Ctrl, we assigned their median samples to the high class. In other words, based on how the Ctrl data points have been split, the median samples of the other datasets would be considered as high.

Verbal Working Memory (VerbalWM)

For VerbalWM, we use two different measures that come from the OSPAN test (VerbalWM_longest and VerbalWM_word). The first measure (VerbalWM_longest) is the maximum verbal working memory capacity, and we use that to split our samples into two groups. This criterion measures the maximum number of words that one can hold in their working memory which ranges from 2 to 6 (6 being the best) [50]. To fine-tune our split, we use the second measure from the OSPAN test, called VerbalWM_word, which measures the total number of words that one gets correct in the test. VerbalWM_word is not a measure of capacity, but rather the overall levels of success in leveraging the VerbalWM [50]. The online test can be found at [54].

VerbalWM_longest

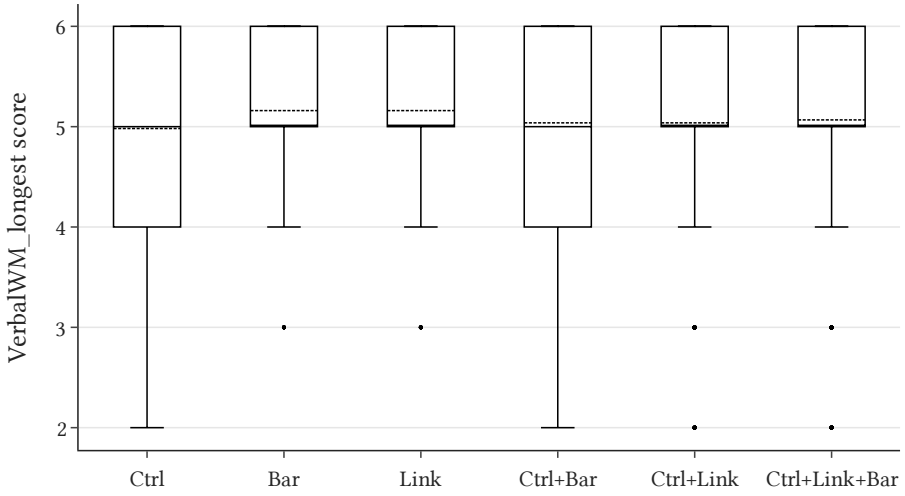

Figure 7: VerbalWM_longest score box plots

Table 4: VerbalWM_longest score stats

	Ctrl (54)	**Bar** (49)	**Link** (26)	**Ctrl+Bar** (103)	**Ctrl+Link** (80)	**Ctrl+Link+Bar** (129)
Mean (SD)	4.98 (1.1)	5.3 (0.8)	5.2 (0.8)	5.1 (0.97)	5.1 (1.0)	5.1 (0.97)
Median (M)	5	5	5	5	5	5
<M | = | >M	15 | **17** | 22	7 | **20** | 22	4 | **12** | 10	22 | **37** | 44	19 | **29** | **32**	26 | **49** | 54

VerbalWM_word

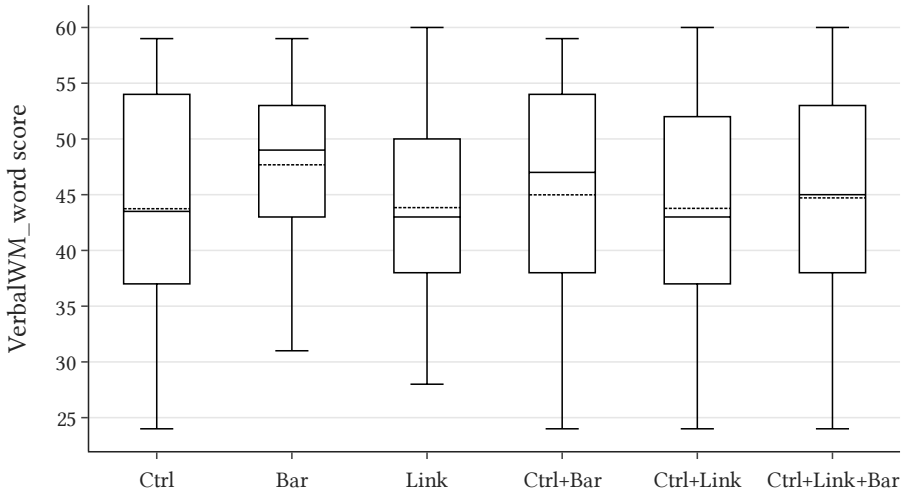

Figure 8: VerbalWM_word score box plots

Table 5: VerbalWM_word score stats

	Ctrl (54)	**Bar** (49)	**Link** (26)	**Ctrl+Bar** (103)	**Ctrl+Link** (80)	**Ctrl+Link+Bar** (129)
Mean (SD)	43.7 (10.3)	47.4 (8.5)	44.2 (7.8)	45.5 (9.6)	43.9 (9.5)	44.7 (9.2)
Median (M)	43.5	48	48	47	43.5	45
<M | = | >M	27 | 0 | 27	23 | 2 | 24	13 | 0 | 13	50 | 5 | 48	40 | 0 | 40	62 | 3 | 64

VerbalWM fine-tuned split:

Table 6: VerbalWM split

	Ctrl (54)	**Bar** (49)	**Link** (26)	**Ctrl+Bar** (103)	**Ctrl+Link** (80)	**Ctrl+Link+Bar** (129)
<M | >M	15+**10** | 22+**7**	7+**14** | 22+**6**	4+**8** | 10+**4**	22+**26** | 44+**11**	19+**18** | 32+**11**	26+**31** | 54+**18**

The median split on the VerbalWM_longest scores resulted in a large number of median samples for all datasets (Table 4). Hence, the simple strategy of assigning these people to either the low or the high class would not yield a good balance. To achieve a more balanced distribution, we used a secondary measure called VerbalWM_word (Table 5) to further split these median samples [1]. Table 6 shows how the median samples from Table 4 were split and assigned to each class based on the VerbalWM_word measure.

Task Time

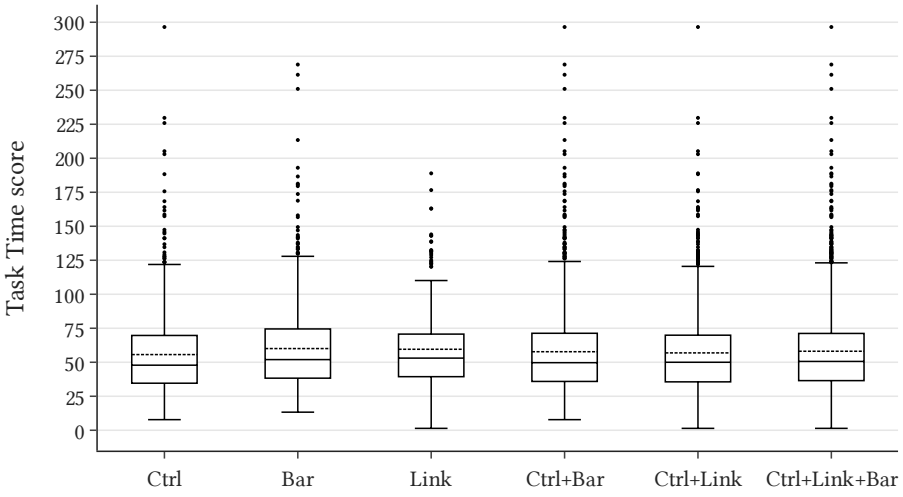

Figure 9: Task Time score box plots

Table 7: Task Time score stats

	Ctrl (756)	**Bar** (686)	**Link** (350)	**Ctrl+Bar** (1442)	**Ctrl+Link** (1106)	**Ctrl+Link+Bar** (1792)
Mean (SD)	55.6 (32.1)	60.1 (32.9)	59.6 (29.8)	57.8 (32.6)	56.9 (31.4)	58.1 (32.0)
Median (M)	47.9	51.95	53.1	49.7	50	50.6
<M | = | >M	378 | 0 | 378	343 | 0 | 343	175 | 0 | 175	720 | 2 | 720	552 | 4 | 550	894 | 4 | 894

For task time, the median split provided a perfect class balance for each of the three study datasets. In the combined datasets, we ended up with only a few median samples that, based on the median of the Ctrl dataset, we assigned to the high class.

Task Accuracy

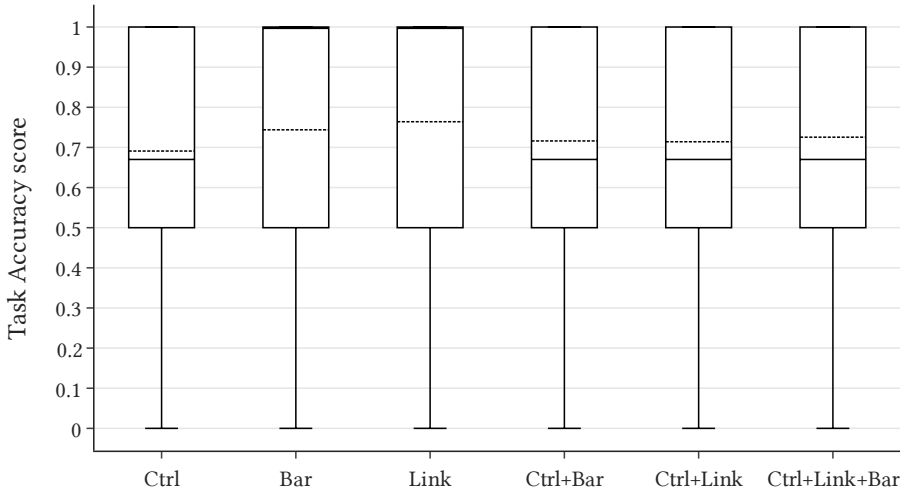

Figure 10: Task Accuracy score box plots

Table 8: Task Accuracy score stats

	Ctrl (756)	**Bar** (686)	**Link** (350)	**Ctrl+Bar** (1442)	**Ctrl+Link** (1106)	**Ctrl+Link+Bar** (1792)
Mean (SD)	0.69 (0.30)	0.74 (0.30)	0.76 (0.29)	0.72 (0.30)	0.71 (0.30)	0.73 (0.30)
Median (M)	0.67	1	1	0.67	0.67	0.67
<M | = | >M	283 | 161 | 312	335 | 351 | 0	159 | 191 | 0	507 | 272 | 663	383 | 220 | 503	607 | 331 | 854

Table 9: Task Accuracy split

	Ctrl (756)	**Bar** (686)	**Link** (350)	**Ctrl+Bar** (1442)	**Ctrl+Link** (1106)	**Ctrl+Link+Bar** (1792)
< 1 | = 1	444 | 312	335 | 351	159 | 191	779 | 663	603 | 503	938 | 854

In all three studies, participants were asked three comprehension questions at the end of each task. The proportion of questions answered correctly was used as their task accuracy score. In the Ctrl dataset, the median accuracy was 0.67 (i.e., getting two out of three questions right). In the adaptive datasets, however, more than half of the participants were able to get all three questions right. As a result, the median score for these datasets was 1 (Table 8). Due to the special distribution of task accuracy scores, instead of taking the median split approach, we decided to assign all participants who scored 1 to the high class and everyone else, who scored less than that, to the low class (Table 9).

Table 10 summarizes the resulting class distributions for each target measure after performing the splits on each dataset.

Table 10: Class distributions

		Ctrl	Bar	Link	Ctrl+Bar	Ctrl+Link	Ctrl+Link+Bar
VisLit	low	0.54	0.57	0.54	0.50	0.54	0.51
	high	0.46	0.43	0.46	0.50	0.46	0.49
ReadP	low	0.50	0.49	0.46	0.48	0.49	0.50
	high	0.50	0.51	0.54	0.52	0.51	0.50
VerbalWM	low	0.46	0.43	0.46	0.47	0.46	0.44
	high	0.54	0.57	0.54	0.53	0.54	0.56
Task Time	low	0.50	0.50	0.50	0.50	0.50	0.50
	high	0.50	0.50	0.50	0.50	0.50	0.50
Task Accuracy	low	0.59	0.51	0.45	0.54	0.54	0.52
	high	0.41	0.49	0.55	0.46	0.46	0.48

4.2. Eye-Tracking Features for Prediction

Table 11: Summary of eye tracking features [1]

a) Overall Gaze Features (21)
- Fixation rate & Fixation duration (mean, SD, Max) - Saccade Duration, Distance, & Velocity (mean, SD, max) - Absolute and Relative saccade angles (mean, SD, rate)
b) Pupil width and Head distance Features (14)
- Pupil width / head distance (mean, SD, min, max) - Pupil width / head distance at first and last recorded fixation - Pupil dilation velocity (mean, SD, min, max)
c) AOI Features (34 x 7 AOI = 238)
- Fixation Rate, Fixation duration in AOI (mean, SD, max) - Time to first fixation in AOI - Proportion of time, Proportion of fixations in AOI - Number, Prop. of transitions from this AOI to every AOI - Pupil width, pupil change velocity, and head distance when looking at AOI (mean, SD, min, max)

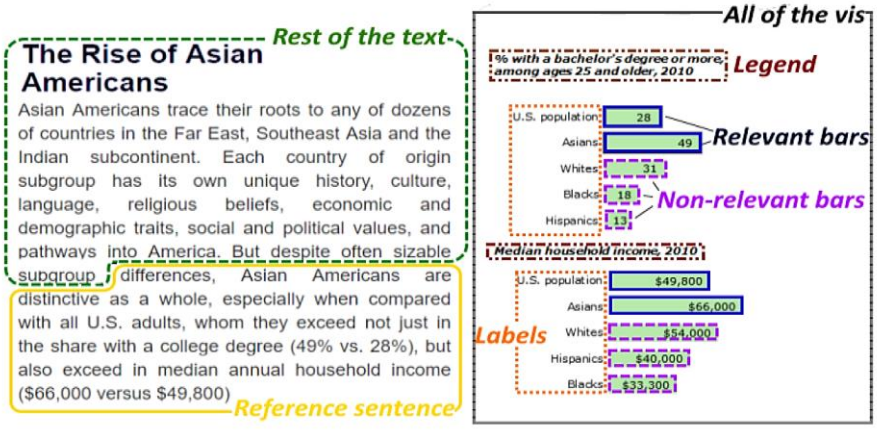

Figure 11: Set of AOIs defined over a sample MSNV [1]

The EMDAT[1] Python library was used for processing the raw eye-tracking data. EMDAT generates a set of eye-tracking features based on the gaze captured over the entire display, as well as over specific regions known as Areas of Interest (AOIs). As shown in Figure 11, we broke our MSNVs into 7 AOIs to encompass the main elements in the text component (i.e., reference sentences, and remaining text), and in the visualization (i.e., relevant bars, non-relevant bars, legends, labels, and all of the vis). These AOIs enabled capturing the gaze processing generated by each modality, as well as gaze transitions among the modalities and the main elements within each.

- **Gaze features** (Table 11.a) were computed using the fixations (gaze maintained at one point on the screen) and saccades (quick eye movement between two fixations).
- **Pupil and head distance features** (Table 11.b) were computed using the pupil size (baseline-adjusted; Iqbal et al. 2005) and distance from both eyes to the screen.
- **Areas of interest (AOI) features** (Table 11.c) were computed using the gaze, pupil and head distance features over each of the salient regions (AOIs) in the MSNV, as well as transitions between them.

These features are similar to the ones used in related user modeling work in InfoVis [31][16] and HCI [15][51][52][53]. We also consider AOI specific pupil dilation and head distance to the screen features (last row in Table 11.c) as used in [1].

4.3. Machine Learning Setup

Within-task prediction

Following [31][16], we investigate at which point within a task, in the control condition, we can best predict our users' cognitive abilities and task performance (*target measures* from now on) to gauge how early adaptive interventions could be triggered based on these predictions. We consider each task as an independent data point. We evaluate prediction accuracy after each second of interaction within a given task, by feeding to the classifiers the corresponding eye tracking data, up to 29 seconds. We stop at 29 seconds because this represents half of the average task time in our control dataset, and making predictions to trigger adaptive support beyond this point would not be very timely in an online scenario. For within task predictions, the classifiers are trained on data from all the individual tasks in the training set, regardless of the order in which they were performed. To make predictions at each time t (from 1 to 29 seconds) for task n, a classifier is trained on feature vectors built from user gaze data from the beginning up to time t, for all the other tasks in the training set, and prediction for task n is made by feeding to the classifier the feature vector built from the corresponding gaze data from 0 to t. [1]

Across-tasks prediction

Predictions within-task are relevant to predict task performance for that task, as well as cognitive abilities in a situation when a user is just performing that one task. However, cognitive abilities usually do not

[1] Eye Movement Data Analysis Toolkit (EMDAT) is a library for processing eye gaze data, developed in the University of British Columbia (github.com/ATUAV/EMDAT)

change in the short term. In our case, a given user has the same level of reading ability, vis literacy and verbal WM across all tasks. Thus, we also explore how accuracy in predicting these user characteristics evolves over time across tasks, namely as users work through a sequence of 14 tasks, as they did in the original control user study [39]. For the across tasks prediction, we evaluate prediction accuracy at the end of each task within a sequence. For making across tasks predictions at the end of the nth task in sequences, the feature vector is built from user gaze data during this task and all its preceding tasks in the sequence. The classifiers are trained on eye-tracking data from the task sequences in the training set, looking at data from the beginning of the first task to the end of nth task in each sequence. [1]

For both within task and across tasks classification, we use the best performing models reported in [1]. That is, we use Logistic Regression (LR) for predicting all target labels within-task, Random Forest (RF) for predicting VisLit and ReadP across tasks and eXtreme Gradient Boosting (XGB) for VerbalWM across tasks. In [1], we used the classification algorithms available in the Caret package [48] in R. However, in this work, we will use the scikit-learn package [49] in Python as this is a more versatile library for training machine learning models. For a consistent comparison, we reproduce the control results from [1] using this library (more details in the appendix).

To train the classifiers for each time-step (29 time-steps within-task, and [14] time-steps across-tasks), we perform 10-fold cross-validation over users, so that users in the test fold never appear in the training fold. Predictions for users in the test fold at a given time step are solely made based on classifiers trained on users in the train fold at the exact same time step, so as to use eye-tracking features built over the same amount of interaction [1]. Highly correlated features (above 0.85) were removed based on the train folds in each iteration [1]. The cross-validation process is repeated 10 times (runs) for reproducibility purposes. We report classification performance in terms of accuracy averaged over the 10 folds and the 10 runs, and use a majority class classifier as a baseline. [1]

4.4. Combination Strategies and Evaluation

In this section, we will describe the different strategies used for combining the additional gaze data from the adaptive studies (Bar and Link) with our control data (Ctrl). In all of our combination strategies, we evaluate the models only on the control data. We do this because we are primarily interested in the performance of the models in the control condition with no interventions used as this will be the condition in which the models would ultimately be deployed to make predictions that guide provision of help. In the combination of each adaptive dataset with our control data for training, for consistency, labeling of the data points is accomplished by performing the median split on the union of all participants from both the control and the designated adaptive study because we want the same score to represent the same label in both studies (e.g., a VisLit score of 37 would be considered low in both studies). For testing, we split the control data using the labels generated by the median split on the combination and use the majority-class thus obtained as our baseline accuracy.

In our naming convention, we identify our models by the data on which they were trained and the combination strategy used. For instance, Bar refers to the model trained on the Bar data only and tested on control, Ctrl+Bar refers to the model trained on the combination of control and bar data and tested on control, etc.

4.4.1. Training on Each Adaptive Dataset Alone: Bar, Link

First, in order to evaluate the predictive knowledge that our model could gain from each adaptive dataset alone for making predictions in the control condition, we train the model on each of the Bar and the Link dataset, separately, and evaluate their performance on the control data. Bar refers to the model trained on the Bar dataset, and Link refers to the model trained on the Link dataset. For each adaptive dataset, after performing the median split on its combination with Ctrl, the control data points are separated from the adaptive ones and the model is trained only on the adaptive data points.

4.4.2. Naïve Combination of Adaptive and Control Datasets: Ctrl+Bar, Ctrl+Link

In this strategy, for each adaptive dataset, we simply combine the entirety of the adaptive dataset, including all participants and all feature values, with the Ctrl dataset, and evaluate the performance of our model when trained on this combination.

For this purpose, we perform a 10-fold cross-validation on the Ctrl dataset. In each iteration of the cross-validation, all of the examples from the adaptive dataset will be added to the train fold. The model is trained on the combined data and then evaluated on the Ctrl examples in the test fold. The cross-validation is not performed on the combination of the two datasets because we do not want any of the adaptive examples to end up in the test folds.

4.4.3. Other Combination Strategies

In machine learning, we always assume that our training set is comprised of data points (X, y) that are independently drawn from the same distribution $p(X, y)$ with $p(X, y) = p(y|X)p(X)$, and $p(y|X)$ being the relationship that we want the model to learn. This is known as the IID assumption (independent and identically distributed). If our training data do not come from the same distribution as our test data, our model will not be able to capture, from the training data, the desired relationship that describes the test data and, therefore, would not perform well. Since the normal gaze behavior of the participants, in the adaptive studies, were impacted by the presence of interventions, we cannot safely assume that the adaptive data come from the same distribution as our control data. That is, $p(X_{control}, y) \neq p(X_{adaptive}, y)$ where $X_{control}$ is the normal gaze behavior in the control condition, and $X_{adaptive}$ is the influenced gaze behavior as a result of interventions. For that reason, a naïve combination would likely violate the IID assumption.

In hopes of a more effective learning from the combined data, in the following strategies[2], we attempt to mitigate the impact of interventions to make the adaptive datasets more similar to the control. By doing this, we intend to take advantage of those aspects of the adaptive dataset that are already similar to the Ctrl, which can help us, and avoid those affected by the interventions, which could potentially confuse the model and work to our disadvantage.

[2] For each measure, we also tried combining with Ctrl, only the high group from the Bar dataset as this group did not benefit from the interventions as much as the low group did. However, this strategy did not yield satisfying results. For that reason, we will not cover this approach.

4.4.3.1 Excluding the Affected Features (EAF): Ctrl+Bar (EAF), Ctrl+Link (EAF)

A basic way to make the adaptive datasets more similar to Ctrl is to remove all the features whose values were affected due to the interventions. We consider a feature to be affected if the mean difference between its control and adaptive values is significant. Conversely, a feature with an insignificant mean difference across the two studies is considered to have remained unaffected (i.e., interventions had very little or no influence on that feature). Essentially, our goal, in this strategy, is to train the model on the entire combined data using only the unaffected features. For this purpose, in each iteration of the cross-validation, for each individual feature, we conduct an independent t-test between the control and the adaptive samples in the train set. We do this for each class separately because class, itself, is a confounding factor and also interventions could have affected different features in each class. We use a significance threshold of 0.05. After running the tests, for each class, we might end up with a different set of affected features. We exclude the features in the union of these two sets from our train and test sets and train/evaluate our model based only on the remaining features (i.e., the so-called unaffected features).

We acknowledge that this strategy, for the sheer purpose of mitigating the impact of interventions, could potentially deprive the model of some highly predictive features. However, it could also serve as a feature selection mechanism. As a result, depending on whether the affected features were useful or not, this approach could either help or severely hurt our performance.

4.4.3.2 Imputing the Affected Features (IAF): Ctrl+Bar (IAF), Ctrl+Link (IAF)

In the previous strategy, in an attempt to mitigate the differences between the Ctrl and the adaptive datasets, the affected features were completely excluded from the training process. This would yield a consistent combination. However, it could also sacrifice potentially useful features whose control values would have benefitted the training of our model. Therefore, in order to take advantage of the entire features and, at the same time, reduce the mentioned differences, we came up with a more forgiving combination strategy. In this approach, instead of removing the affected features from the Ctrl and the adaptive datasets, we would eliminate the impact of interventions by imputing (replacing) their adaptive values with their mean control values (from Ctrl). For this purpose, similar to the previous strategy, in each iteration of the cross-validation, we use independent t-tests to identify the significantly affected features in each class. After that, for each class separately, using only the training samples, we substitute the values of those features in the adaptive dataset with their mean control values from Ctrl.

The advantage of this approach is that while neutralizing the negative impact of the features whose values were significantly affected by the interventions in the adaptive datasets, it still allows the entire feature values from Ctrl to influence the training.

5. Results

In this section, we will first report our results on the within-task predictions for the cognitive abilities and task performance measures (section 5.1), followed by results on across-tasks predictions (section 5.2)

For each target measure, we first provide performance results of each of the datasets alone (4.4.1). We then report the results of combining each of the adaptive datasets with the control dataset using our combination strategies (section 4.4.2, 4.4.3).

5.1. Within Task Prediction Results

We evaluate the performance of our strategies by running a one-way repeated measures ANOVA with accuracy as the dependent variable, strategy type as the factor, and time step (29) as the repeated measure. To account for running multiple ANOVAs, we adjust all obtained p values using the Benjamini Hochberg [47] method to control for the false discovery rate (FDR). Statistical significance is reported at $p < 0.05$ after adjustment.

5.1.1. VisLit

a. Training on each dataset alone and evaluating on Ctrl

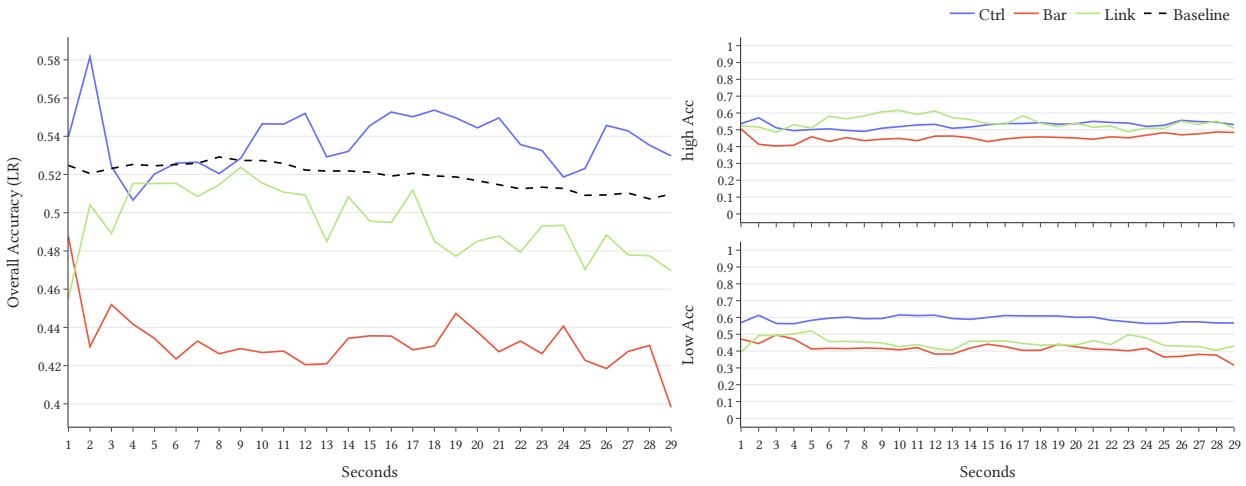

Figure 12: Results for VisLit LR over different windows within-task when trained on each dataset alone

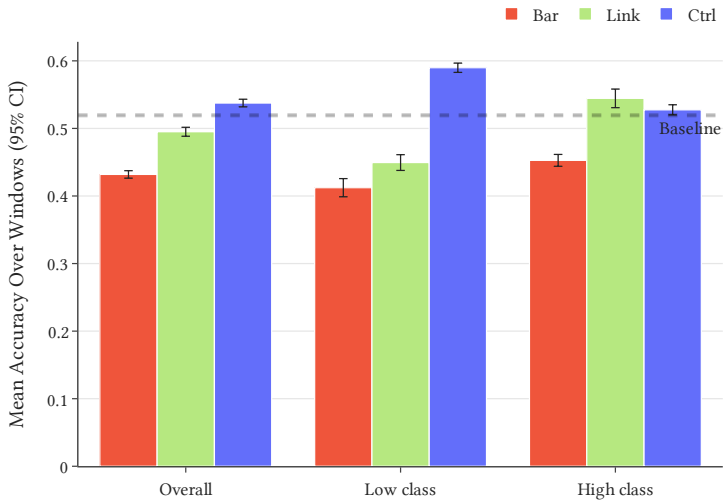

Figure 13: Average performance over all windows within-task for VisLit LR when trained on each dataset alone

Table 12: VisLit LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.006)	-	-
Ctrl	0.54 (0.015)	0.59 (0.018)	0.53 (0.020)
Bar	0.43 (0.014)	0.41 (0.035)	0.45 (0.023)
Link	0.50 (0.017)	0.45 (0.031)	0.54 (0.036)

Figure 13 shows the performance of the LR classifier at predicting VisLit when trained on each dataset alone. We compare these results by running post-hoc pairwise comparisons (t-tests adjusted for FDR) on overall performance. The results of the comparisons show: *Ctrl* > *Baseline* > *Link* > *Bar* (">" indicates statistically significant difference). Both the Bar ($p < 0.0001$, $\eta^2 = 0.93$)[3] and the Link dataset ($p < 0.0001$, $\eta^2 = 0.63$) provide lower performance than Ctrl and the baseline.

[3] $\eta 2 \leq 0.06$ indicates a small effect; $0.06 < \eta 2 < 0.14$ indicates a medium effect; $0.14 \leq \eta 2$ indicates a large effect

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

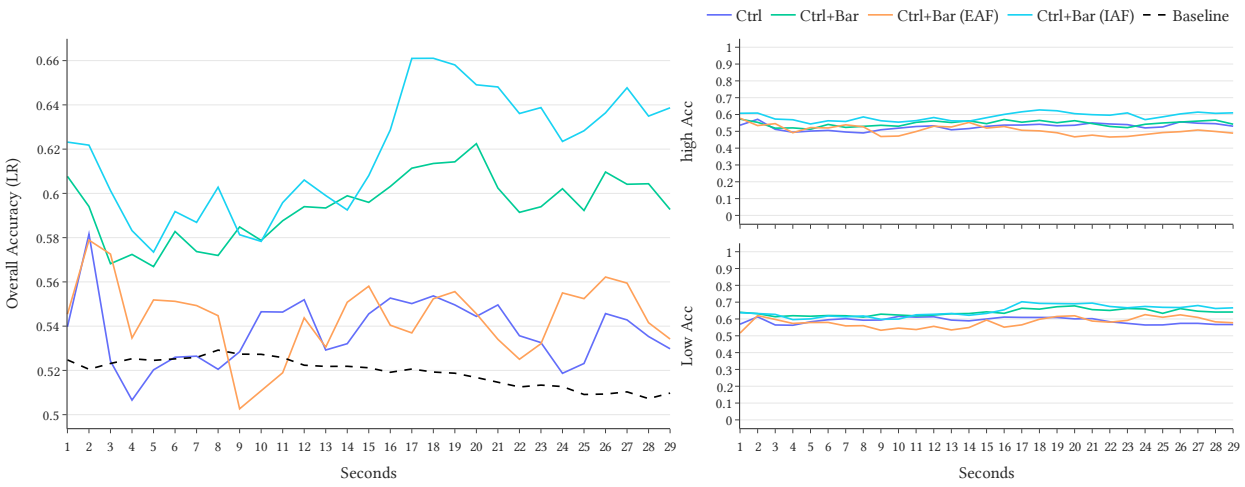

Figure 14: Results for VisLit LR over different windows within-task when trained on the combination of Ctrl and Bar

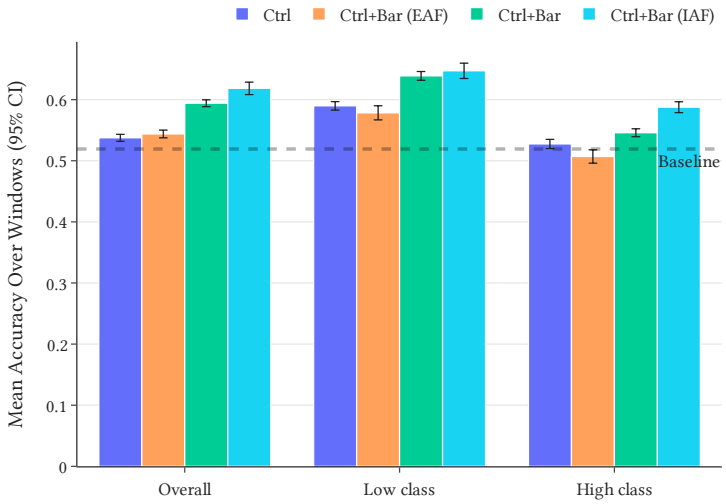

Figure 15: Average performance over all windows within-task for VisLit LR when trained on the combination of Ctrl and Bar

Table 13: VisLit LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.006)	-	-
Ctrl	0.54 (0.015)	0.59 (0.018)	0.53 (0.020)
Ctrl+Bar	0.59 (0.015)	0.64 (0.019)	0.55 (0.017)
Ctrl+Bar (EAF)	0.54 (0.017)	0.58 (0.030)	0.51 (0.029)
Ctrl+Bar (IAF)	0.62 (0.027)	0.65 (0.033)	0.59 (0.023)

Figure 15 shows the performance of the LR classifier at predicting VisLit when trained on the combination of Ctrl and Bar. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Bar (IAF)* > *Ctrl+Bar* > *Ctrl+Bar (EAF)* = *Ctrl* > *Baseline* (">" indicates statistically significant difference, while "=" indicates statistical equivalence). The naïve combination of Ctrl and Bar (Ctrl+Bar) outperforms the control dataset alone (Ctrl) ($p < 0.0001$, $\eta^2 = 0.78$). The EAF strategy was intended to mitigate the differences between the two datasets. However, not only did it not provide any improvement over the naïve combination, but it also performed worse ($p < 0.0001$, $\eta^2 = 0.72$). This suggests that gaze features, which were significantly affected due to the bar interventions, were highly predictive of VisLit, for their exclusion resulted in a drop in performance despite making the combination more consistent. The IAF strategy, on the other hand, proved effective in maximizing the gain from the combined dataset with a significant improvement over the naïve combination ($p < 0.0001$, $\eta^2 = 0.24$). Inspection of class performance reveals that this improvement is mostly due to an increase in the accuracy of the high-class. IAF reached its peak at window 17 with an overall accuracy of 0.66.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

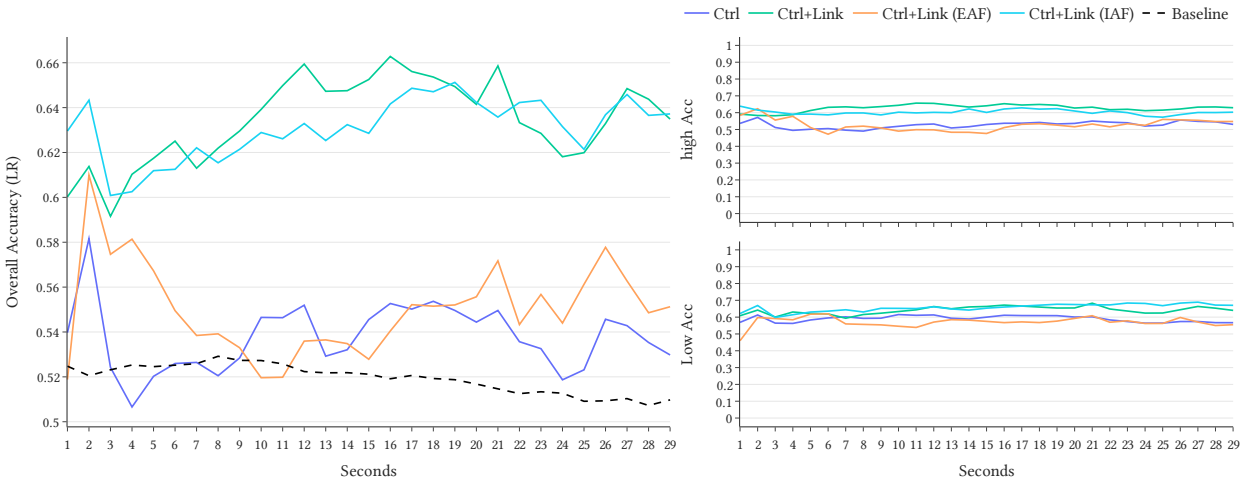

Figure 16: Results for VisLit LR over different windows within-task when trained on the combination of Ctrl and Link

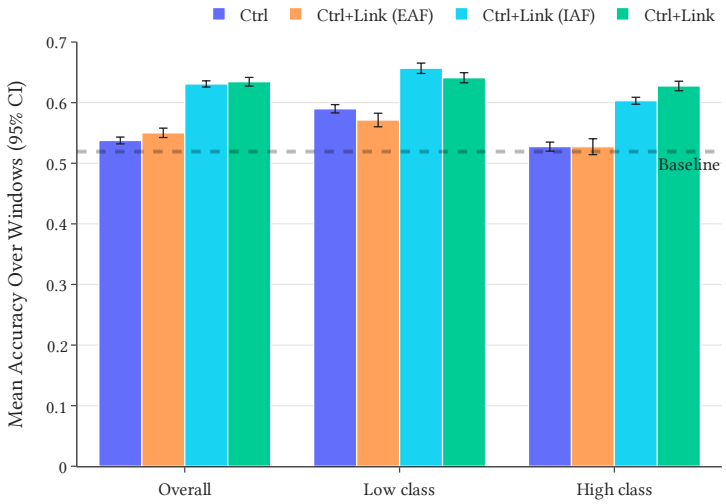

Figure 17: Average performance over all windows within-task for VisLit LR when trained on the combination of Ctrl and Link

Table 14: VisLit LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.006)	-	-
Ctrl	0.54 (0.015)	0.59 (0.018)	0.53 (0.020)
Ctrl+Link	0.63 (0.019)	0.64 (0.022)	0.63 (0.021)
Ctrl+Link (EAF)	0.55 (0.020)	0.57 (0.029)	0.53 (0.035)
Ctrl+Link (IAF)	0.63 (0.013)	0.66 (0.023)	0.60 (0.015)

Figure 17 shows the performance of the LR classifier at predicting VisLit when trained on the combination of Ctrl and Link. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link = Ctrl+Link (IAF) > Ctrl+Link (EAF) = Ctrl > Baseline*. The naïve combination of Ctrl and Link (Ctrl+Link) outperforms Ctrl ($p < 0.0001$, $\eta^2 = 0.89$). The EAF strategy resulted in a lower performance than the naïve combination ($p < 0.0001$, $\eta^2 = 0.89$). It appears that link interventions, similar to the bar interventions, affected features useful for predicting VisLit. The IAF strategy did not help either and yielded statistically similar results to the naïve combination ($p = 0.40$, $\eta^2 = 0.01$). In fact, IAF increased the gap between class accuracies. This indicates that there was a degree of knowledge to be gained from the affected link features, for imputing their values hampered this however small gain.

Table 15: VisLit LR Best performing windows (within-task)

Strategy	Window	Type	Overall Acc	Low Acc	High Acc
Ctrl	2 sec	Window with highest overall acc	0.58	0.61	0.57
Ctrl+Link (IAF)	2 sec	Window with highest overall acc	0.64	0.67	0.62
Ctrl+Link	12 sec	Window with highest overall acc	0.66	0.66	0.66
Ctrl+Bar (IAF)	17 sec	Window with highest overall acc	0.66	0.70	0.62

Table *13*Table 15 reports the accuracy of our best performing strategies, namely strategies that performed better than Ctrl, either by having higher overall accuracy over time or having more balanced class accuracies at their viable window(s). For each strategy, we report the window with the highest accuracy. If this comes quite late in our 29 second interval, we look to see if there is an earlier window with acceptable accuracy and report that as well (this does not apply for Table 15 because all best windows came early enough in the interaction). The peak performance of Ctrl was attained at window 2 with an overall accuracy of 0.58 (see Figure 12). However, at that same window, Ctrl+Link (IAF) achieves an accuracy of 0.64. Ctrl+Link yields the highest accuracy of 0.66 at window 12 with perfectly balanced class performance (see Figure 16). We leave it to the researchers to decide whether a 2% increase in accuracy justifies the longer wait.

In terms of VisLit, the EAF strategy proved ineffective for both adaptive datasets. The IAF strategy improved performance for the combination of Ctrl and Bar, but it did not help with the combination of Ctrl and Link. This is because the original feature values of the Link were more beneficial than those of the Bar which is also consistent with the higher performance of Link compared to Bar when used alone (section 5.1.1.a).

5.1.2. ReadP

a. Training on each dataset alone and evaluating on Ctrl

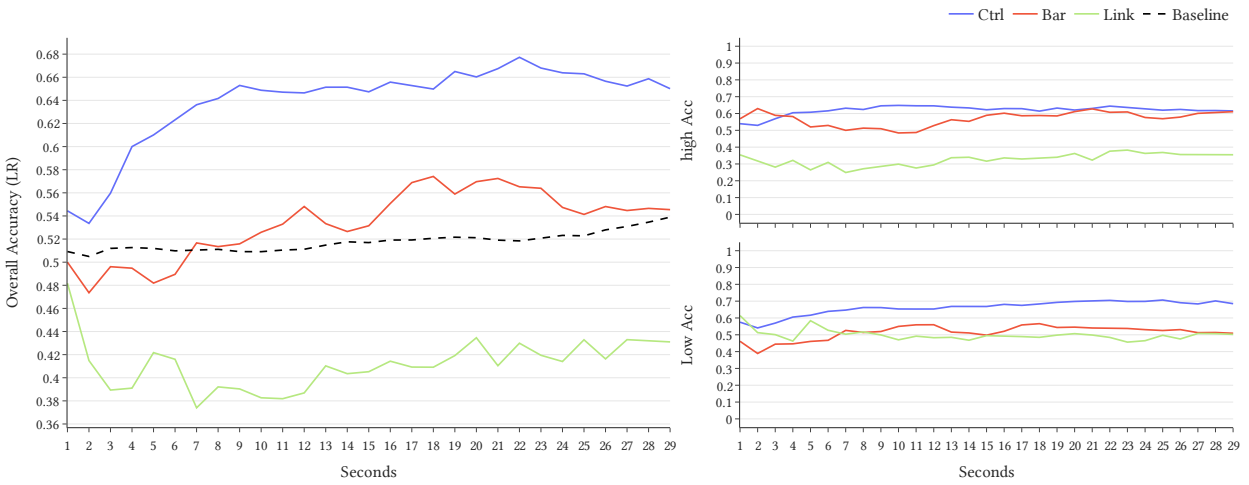

Figure 18: Results for ReadP LR over different windows within-task when trained on each dataset alone

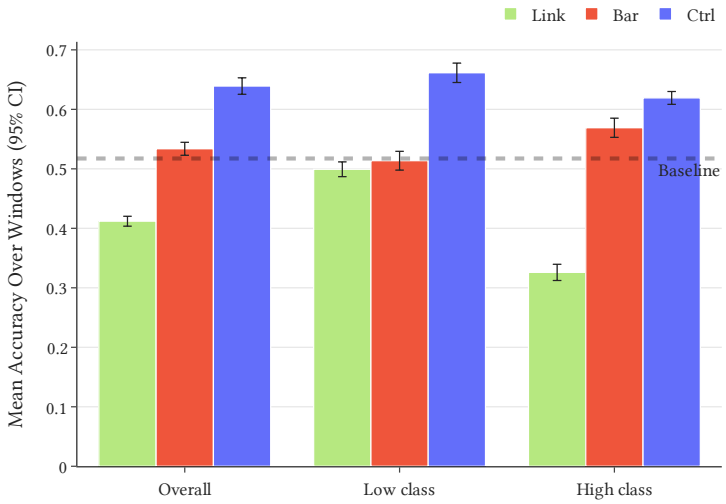

Figure 19: Average performance over all windows within-task for ReadP LR when trained on each dataset alone

Table 16: ReadP LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.008)	-	-
Ctrl	0.64 (0.036)	0.66 (0.043)	0.62 (0.028)
Bar	0.53 (0.029)	0.51 (0.041)	0.57 (0.042)
Link	0.41 (0.022)	0.50 (0.033)	0.33 (0.036)

Figure 19 shows the performance of the LR classifier at predicting ReadP when trained on each dataset alone. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl* > *Bar* > *Baseline* > *Link*. Both the Bar ($p < 0.0001$, $\eta^2 = 0.72$) and the Link dataset ($p < 0.0001$, $\eta^2 = 0.94$) yield lower performance than Ctrl.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

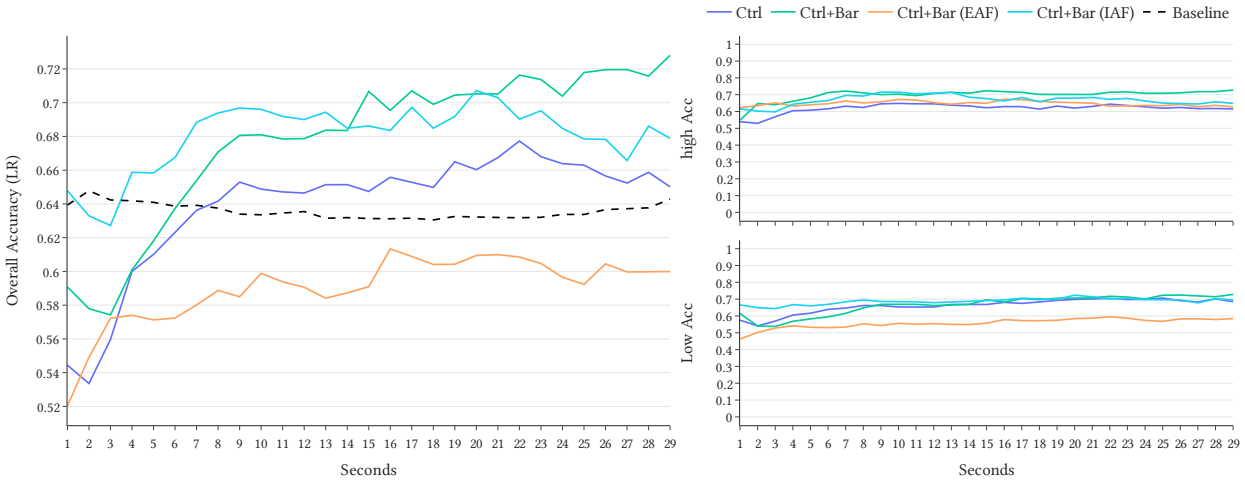

Figure 20: Results for ReadP LR over different windows within-task when trained on the combination of Ctrl and Bar

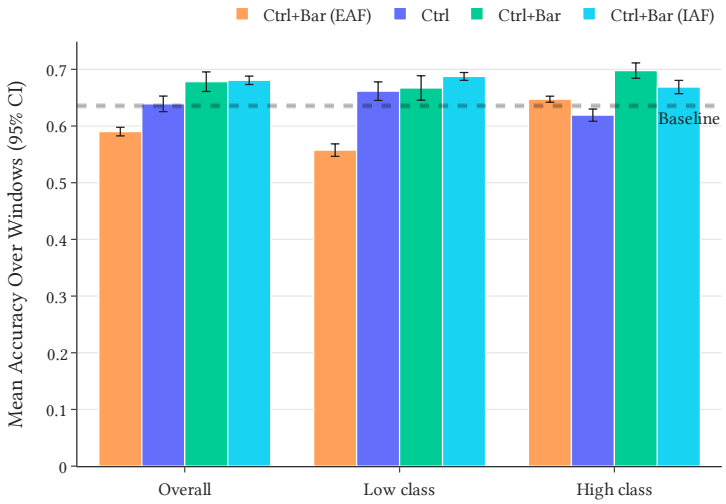

Figure 21: Average performance over all windows within-task for ReadP LR when trained on the combination of Ctrl and Bar

Table 17: ReadP LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.64 (0.004)[4]	-	-
Ctrl	0.64 (0.036)	0.66 (0.043)	0.62 (0.028)
Ctrl+Bar	0.68 (0.045)	0.67 (0.057)	0.70 (0.036)
Ctrl+Bar (EAF)	0.59 (0.020)	0.56 (0.029)	0.65 (0.014)
Ctrl+Bar (IAF)	0.68 (0.020)	0.69 (0.018)	0.67 (0.031)

Figure 21 shows the performance of the LR classifier at predicting ReadP when trained on the combination of Ctrl and Bar. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Bar (IAF) = Ctrl+Bar > Ctrl = Baseline > Ctrl+Bar (EAF).* Ctrl+Bar outperforms Ctrl (p = 0.0007, η^2 = 0.19). The EAF strategy did not provide any improvement over the naïve combination and performed even worse (p < 0.0001, η^2 = 0.61). This indicates the importance of the eye-tracking features, affected by the bar interventions, for predicting ReadP. The IAF strategy did not improve the overall performance either and yielded statistically similar results to the naïve combination (p = 0.79, η^2 = 0.001). Nevertheless, it provided a steadier performance over the windows with a higher accuracy for the low-class which is the group that benefits from adaptive support the most. The IAF strategy was also more accurate at earlier windows (higher accuracy than all other strategies until window 14, see Figure 20). After window 14, the performance of the naïve combination kept increasing and reached its peak accuracy of 0.72 at the last window (window 29). This suggests that the original feature values from the Bar dataset, accumulated past window 14, were quite useful for predicting ReadP when combined with Ctrl.

[4] Combination of Bar and Ctrl yields a relatively high baseline accuracy for ReadP. This is because, as mentioned in section 4.4, for testing we split the control data using the labels generated by the median split on the combination and use the majority-class thus obtained as our baseline accuracy. The median of the combination dataset is higher than that of Ctrl because the median ReadP score for Bar was higher than Ctrl (see Table 3). As a result, after the split, more than half of the Ctrl data points ended up being labelled as low which led to a higher baseline (majority class) accuracy.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

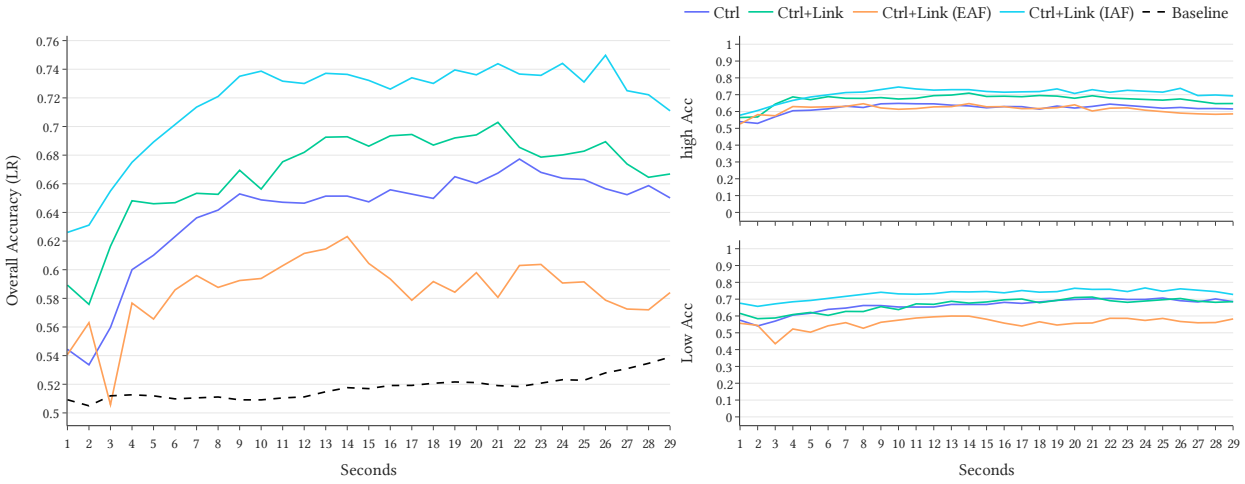

Figure 22: Results for ReadP LR over different windows within-task when trained on the combination of Ctrl and Link

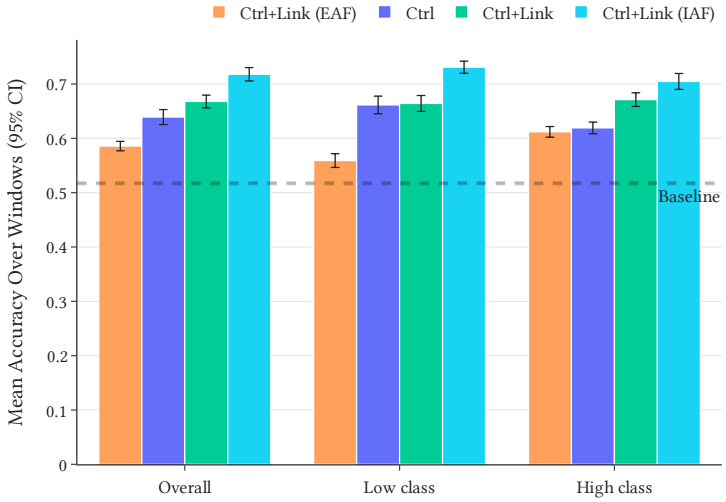

Figure 23: Average performance over all windows within-task for ReadP LR when trained on the combination of Ctrl and Link

Table 18: ReadP LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.008)	-	-
Ctrl	0.64 (0.036)	0.66 (0.043)	0.62 (0.028)
Ctrl+Link	0.67 (0.031)	0.66 (0.038)	0.67 (0.033)
Ctrl+Link (EAF)	0.59 (0.023)	0.56 (0.033)	0.61 (0.026)
Ctrl+Link (IAF)	0.72 (0.032)	0.73 (0.029)	0.70 (0.039)

Figure 23 shows the performance of the LR classifier at predicting ReadP when trained on the combination of Ctrl and Link. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link (IAF) > Ctrl+Link > Ctrl > Ctrl+Link (EAF) > Baseline*. Ctrl+Link outperforms Ctrl ($p = 0.002$, $\eta^2 = 0.15$). The EAF strategy resulted in a lower performance than the naïve combination ($p < 0.0001$, $\eta^2 = 0.70$). It appears that link interventions, similar to the bar interventions, affected features useful for predicting ReadP. The IAF strategy, on the other hand, proved quite effective with a significant increase in performance over to the naïve combination ($p < 0.0001$, $\eta^2 = 0.38$).

Table 19: ReadP LR Best performing windows (within-task)

Strategy	**Window**	**Type**	**Overall Acc**	**Low Acc**	**High Acc**
Ctrl	22 sec	Window with highest overall acc	0.68	0.70	0.64
	9 sec	Earliest window with acceptable overall acc	0.65	0.66	0.65
Ctrl+Bar (IAF)	9 sec	Highest peak in overall acc	0.70	0.69	0.72
Ctrl+Link (IAF)	26 sec	Highest peak in overall acc	0.75	0.76	0.74
	10 sec	Earliest window with acceptable overall acc	0.74	0.73	0.75

Table 19 reports the accuracy of our best performing strategies, namely strategies that performed better than Ctrl, either by having higher overall accuracy over time or having more balanced class accuracies at their viable window(s). For each strategy, we report the window with the highest accuracy. If this comes quite late in our 29 second interval (as it happens for Ctrl and Ctrl+Link (IAF)), we look to see if there is an earlier window with acceptable accuracy and report that as well. The peak performance of Ctrl was attained at window 22 with an overall accuracy of 0.68 (see Figure 18). Ctrl+Link (IAF), however, achieved a higher accuracy of 0.74 at an even earlier window (window 10). The highest accuracy was reached at window 26 by the same model, but this slightly better performance may not justify the longer wait (see Figure 22).

In terms of ReadP, the EAF strategy proved ineffective for both adaptive datasets. The IAF strategy improved performance for the combination of Ctrl and Link. This was also true for the combination of Ctrl and Bar, but only at earlier windows. Due to decreased performance at later windows, IAF was not consistently effective for this combination overall. This strategy was more effective for the Link dataset suggesting that the original feature values of the Link were not as helpful as those of the Bar for predicting ReadP as shown in section 5.1.2.a

5.1.3. VerbalWM

a. Training on each dataset alone and evaluating on Ctrl

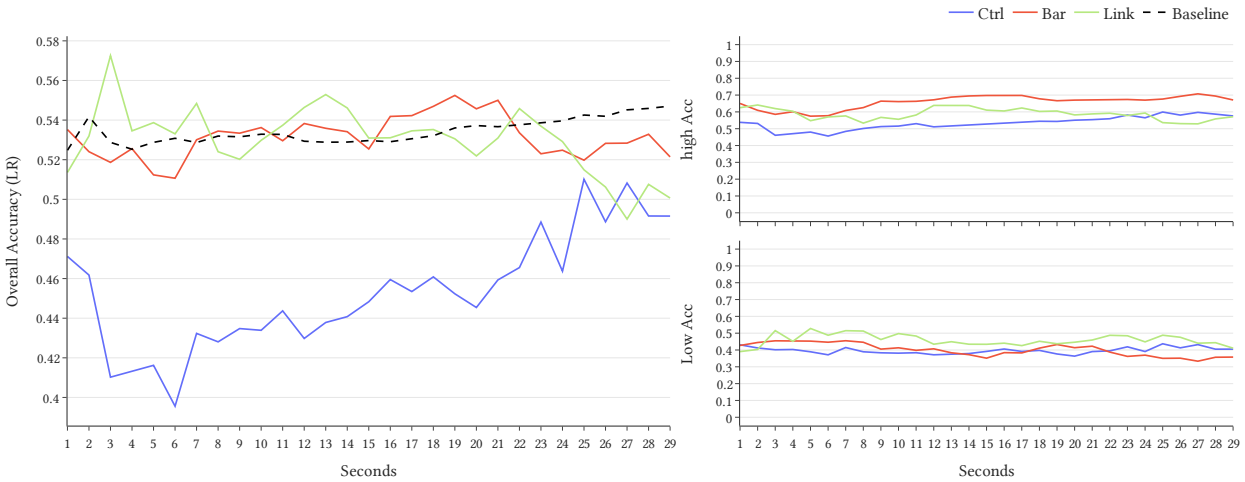

Figure 24: Results for VerbalWM LR over different windows within-task when trained on each dataset alone

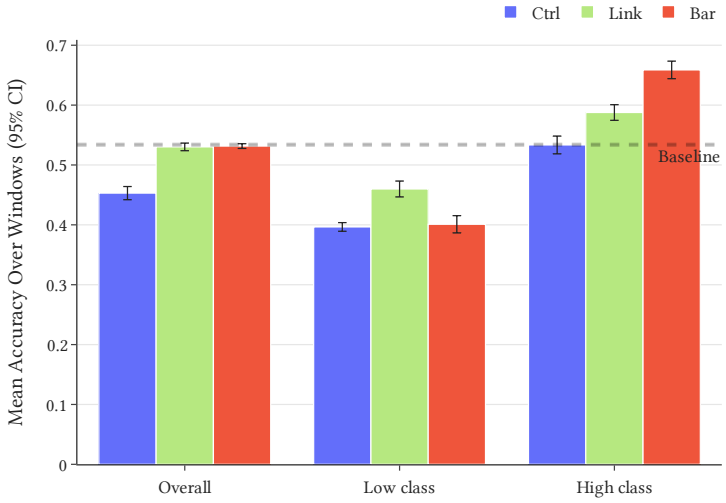

Figure 25: Average performance over all windows within-task for VerbalWM LR when trained on each dataset alone

Table 20: VerbalWM LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.006)	-	-
Ctrl	0.45 (0.029)	0.40 (0.019)	0.53 (0.039)
Bar	0.53 (0.010)	0.40 (0.038)	0.66 (0.039)
Link	0.53 (0.017)	0.46 (0.035)	0.59 (0.034)

Figure 25 shows the performance of the LR classifier at predicting VerbalWM when trained on each dataset alone. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Baseline = Bar = Link > Ctrl.* Ctrl performs significantly lower than the baseline ($p < 0.0001$, $\eta^2 = 0.79$) which was also the case in [1]. Interestingly, both the Bar ($p < 0.0001$, $\eta^2 = 0.76$) and the Link dataset ($p < 0.0001$, $\eta^2 = 0.73$) yield significantly higher performance than Ctrl. Apparently, the classifier learned more from the adaptive datasets than it did from the Ctrl itself in terms of VerbalWM. Notwithstanding, they were not able to beat the baseline either.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

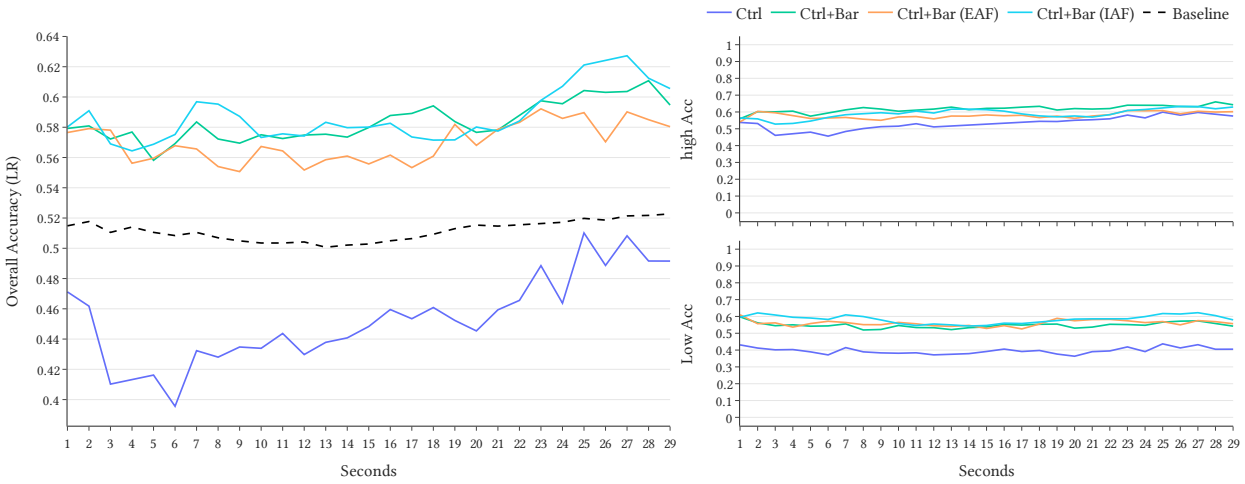

Figure 26: Results for VerbalWM LR over different windows within-task when trained on the combination of Ctrl and Bar

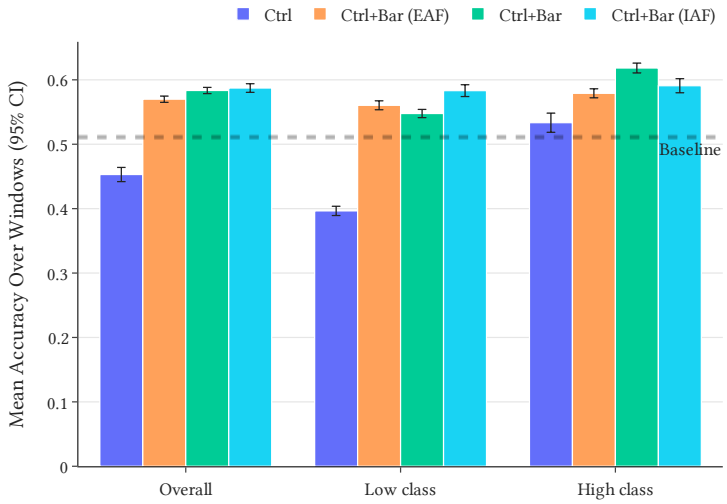

Figure 27: Average performance over all windows within-task for VerbalWM LR when trained on the combination of Ctrl and Bar

Table 21: VerbalWM LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.51 (0.007)	-	-
Ctrl	0.45 (0.029)	0.40 (0.019)	0.53 (0.039)
Ctrl+Bar	0.58 (0.013)	0.55 (0.017)	0.62 (0.020)
Ctrl+Bar (EAF)	0.57 (0.013)	0.56 (0.018)	0.58 (0.018)
Ctrl+Bar (IAF)	0.59 (0.018)	0.58 (0.024)	0.59 (0.029)

Figure 27 shows the performance of the LR classifier at predicting VerbalWM when trained on the combination of Ctrl and Bar. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Bar (IAF) = Ctrl+Bar > Ctrl+Bar (EAF) > Baseline > Ctrl.* Ctrl+Bar outperforms Ctrl ($p < 0.0001$, $\eta^2 = 0.89$). More importantly, it outperforms the previously unbeaten baseline ($p < 0.0001$, $\eta^2 = 0.93$). The EAF strategy did not provide any improvement over the naïve combination and performed worse ($p = 0.0002$, $\eta^2 = 0.22$). The IAF strategy did not improve the overall performance either and yielded statistically similar results to the naïve combination ($p = 0.34$, $\eta^2 = 0.015$). Nevertheless, by improving the accuracy of the low group, it provided more balanced class accuracies.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

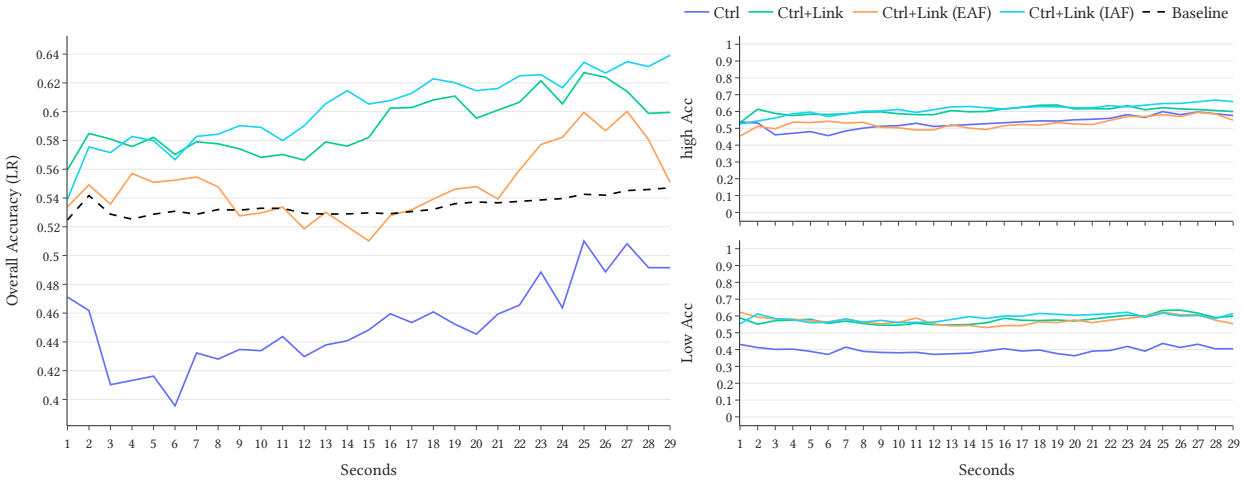

Figure 28: Results for VerbalWM LR over different windows within-task when trained on the combination of Ctrl and Link

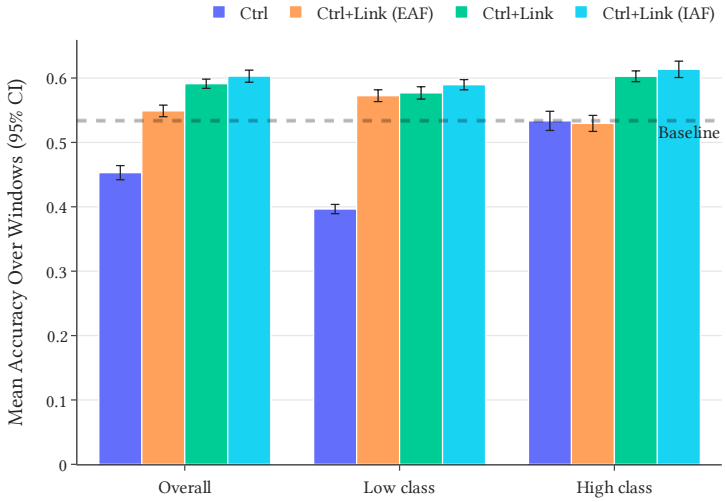

Figure 29: Average performance over all windows within-task for VerbalWM LR when trained on the combination of Ctrl and Link

Table 22: VerbalWM LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.006)	-	-
Ctrl	0.45 (0.029)	0.40 (0.019)	0.53 (0.039)
Ctrl+Link	0.59 (0.019)	0.58 (0.025)	0.60 (0.022)
Ctrl+Link (EAF)	0.55 (0.024)	0.57 (0.024)	0.53 (0.033)
Ctrl+Link (IAF)	0.60 (0.025)	0.59 (0.021)	0.61 (0.034)

Figure 29 shows the performance of the LR classifier at predicting VerbalWM when trained on the combination of Ctrl and Link. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link (IAF) = Ctrl+Link > Ctrl+Link (EAF) > Baseline > Ctrl.* Ctrl+Link outperforms Ctrl ($p < 0.0001$, $\eta^2 = 0.89$) and the unbeaten baseline ($p < 0.0001$, $\eta^2 = 0.80$). The EAF strategy resulted in a lower performance than the naïve combination ($p < 0.0001$, $\eta^2 = 0.49$). The IAF strategy, on the other hand, was more effective and slightly improved performance, though not significantly, ($p = 0.05$, $\eta^2 = 0.07$). The performance of IAF kept increasing and reached a peak accuracy of 0.64 at the last window (window 29).

Table 23: VerbalWM LR Best performing windows (within-task)

Strategy	Window	Type	Overall Acc	Low Acc	High Acc
Ctrl	25 sec	Window with highest overall acc	0.51	0.44	0.60
Ctrl+Bar (IAF)	27 sec	Window with highest overall acc	0.63	0.62	0.63
	7 sec	Earliest window with acceptable overall acc	0.60	0.61	0.58
Ctrl+Link (IAF)	29 sec	Window with highest overall acc	0.64	0.61	0.66
	14 sec	Earliest window with acceptable overall acc	0.61	0.60	0.63

Table 23 reports the accuracy of our best performing strategies at their viable window(s) at predicting VerbalWM. The peak performance of Ctrl was attained at window 25 with an overall accuracy of 0.51 which was still lower than the baseline (see Figure 24). Combination of the adaptive studies provided encouraging results and improved our performance significantly beyond the baseline.

The EAF strategy was not beneficial for either of the adaptive datasets. However, it was less adverse for VerbalWM than it was for the other cognitive abilities. This indicates that the affected gaze features, which were influenced by the adaptive interventions, were, perhaps, not as highly predictive of VerbalWM as they were of the other two cognitive abilities. It also explains why Bar and Link had such high performance on their own (section 5.1.3.a). The insignificant improvement of IAF over the naïve combination can be justified with the same rationale as changing the values of unimportant features should not have a meaningful impact on performance.

5.1.3. Task Time (Speed)

a. Training on each dataset alone and evaluating on Ctrl

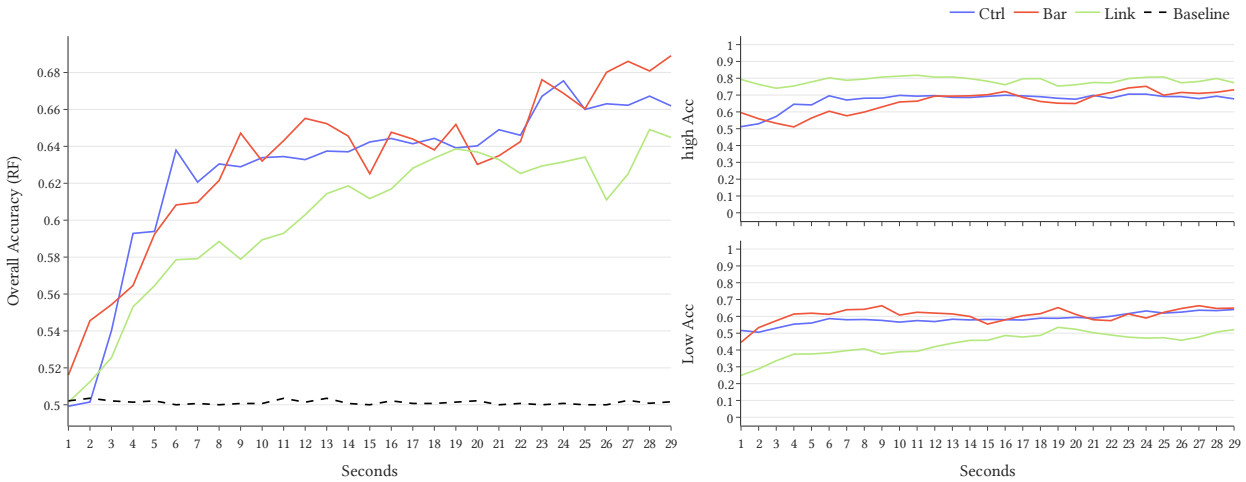

Figure 30: Results for Task Time LR over different windows within-task when trained on each dataset alone

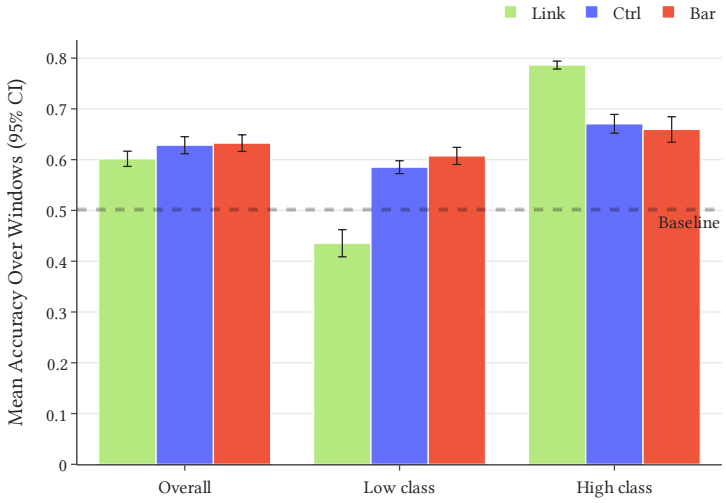

Figure 31: Average performance over all windows within-task for Task Time LR when trained on each dataset alone

Table 24: Task Time LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.50 (0.001)	-	-
Ctrl	0.63 (0.044)	0.59 (0.033)	0.67 (0.049)
Bar	0.63 (0.043)	0.61 (0.044)	0.66 (0.066)
Link	0.60 (0.039)	0.44 (0.070)	0.79 (0.020)

Figure 31 shows the performance of the LR classifier at predicting Task Time when trained on each dataset alone. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Bar = Ctrl > Link > Baseline*. Bar is as predictive as Ctrl with statistically similar results (p = 0.72, η^2 = 0.002). It even provides slightly more balanced class accuracies than Ctrl. Link, on the other hand, was not as predictive as Bar and performed lower than Ctrl (p = 0.02, η^2 = 0.09) with a huge class accuracy gap.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

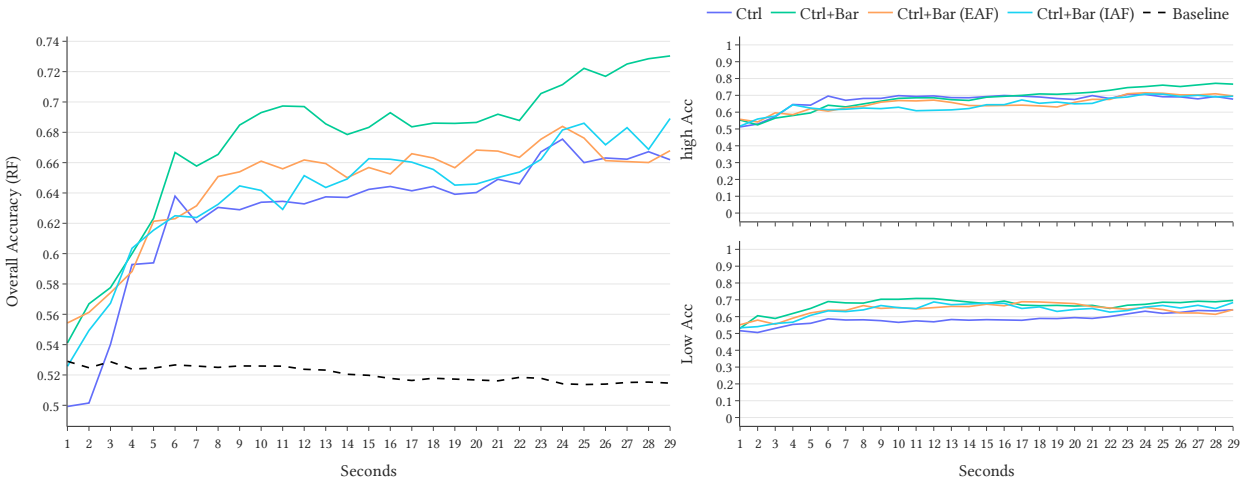

Figure 32: Results for Task Time LR over different windows within-task when trained on the combination of Ctrl and Bar

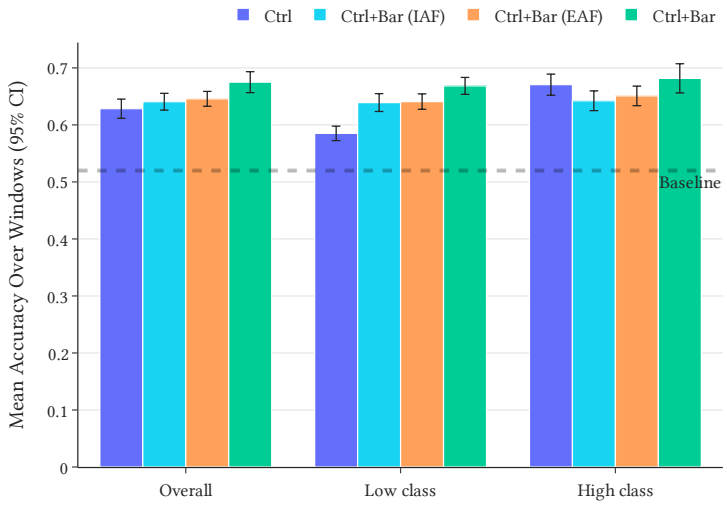

Figure 33: Average performance over all windows within-task for Task Time LR when trained on the combination of Ctrl and Bar

Table 25: Task Time LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.52 (0.005)	-	-
Ctrl	0.63 (0.044)	0.59 (0.033)	0.67 (0.049)
Ctrl+Bar	0.67 (0.048)	0.67 (0.039)	0.68 (0.067)
Ctrl+Bar (EAF)	0.65 (0.034)	0.64 (0.035)	0.65 (0.045)
Ctrl+Bar (IAF)	0.64 (0.039)	0.64 (0.041)	0.64 (0.045)

Figure 33 shows the performance of the LR classifier at predicting Task Time when trained on the combination of Ctrl and Bar. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Bar > Ctrl+Bar (EAF) = Ctrl+Bar (IAF) = Ctrl > Baseline*. Ctrl+Bar outperforms Ctrl ($p = 0.0007$, $\eta^2 = 0.20$). This combination also provides more balanced class performance. The EAF strategy did not provide any improvement over the naïve combination and performed worse ($p = 0.01$, $\eta^2 = 0.11$). The IAF strategy was not beneficial either and performed lower than Ctrl ($p = 0.007$, $\eta^2 = 0.13$), almost as badly as EAF. This is because the Bar dataset alone was already as predictive of Task Time as the Ctrl itself (section 5.1.3.a); therefore, exclusion or imputation of its features deprived the model of potentially useful data.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

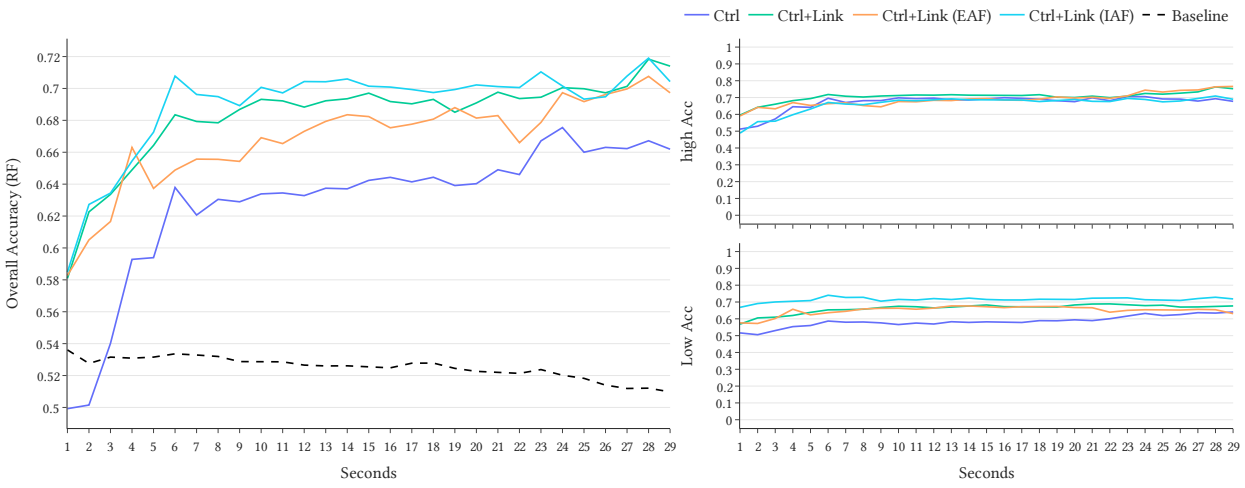

Figure 34: Results for Task Time LR over different windows within-task when trained on the combination of Ctrl and Link

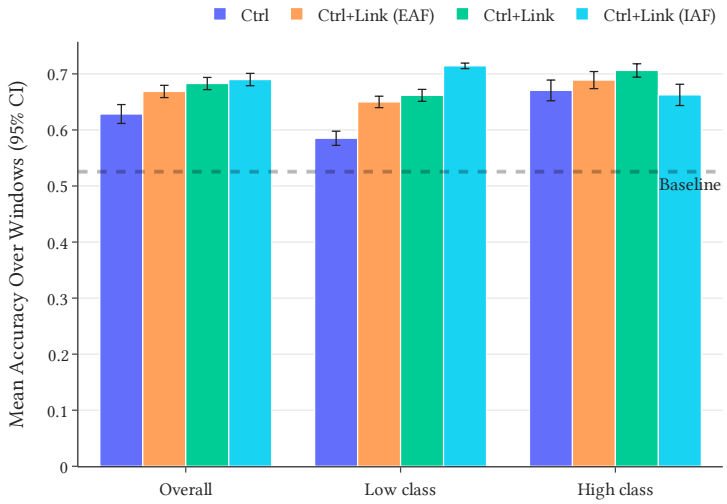

Figure 35: Average performance over all windows within-task for Task Time LR when trained on the combination of Ctrl and Link

Table 26: Task Time LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.007)	-	-
Ctrl	0.63 (0.044)	0.59 (0.033)	0.67 (0.049)
Ctrl+Link	0.68 (0.029)	0.66 (0.028)	0.71 (0.031)
Ctrl+Link (EAF)	0.67 (0.029)	0.65 (0.027)	0.69 (0.040)
Ctrl+Link (IAF)	0.69 (0.029)	0.71 (0.013)	0.66 (0.050)

Figure 35 shows the performance of the LR classifier at predicting Task Time when trained on the combination of Ctrl and Link. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link (IAF) = Ctrl+Link = Ctrl+Link (EAF) > Ctrl > Baseline*. Ctrl+Link outperforms Ctrl ($p < 0.0001$, $\eta^2 = 0.35$). The EAF strategy performed lower than Ctrl+Link, but the difference was not significant ($p = 0.07$, $\eta^2 = 0.06$). The IAF strategy, on the other hand, slightly improved performance. Although this improvement was not significant ($p = 0.4$, $\eta^2 = 0.01$), it provided a higher accuracy for the low class as opposed to the naïve combination (and also Ctrl) where it was the other way around. This is important as the low group is the primary target for personalization.

Table 27: Task Time LR Best performing windows (within-task)

Strategy	Window	Type	Overall Acc	Low Acc	High Acc
Ctrl	24 sec	Window with highest overall acc	0.68	0.63	0.71
	6 sec	Earliest window with acceptable overall acc	0.64	0.59	0.70
Ctrl+Link (IAF)	28 sec	Window with highest overall acc	0.72	0.73	0.71
	6 sec	Earliest window with acceptable overall acc	0.71	0.74	0.67
Ctrl+Bar	29 sec	Window with highest overall acc	0.73	0.70	0.77
	11 sec	Earliest window with acceptable overall acc	0.70	0.71	0.69

Table 27 reports the accuracy of our best performing strategies at their viable window(s) at predicting Task Time. The peak performance of Ctrl was attained at window 24 with an overall accuracy of 0.68 (see Figure 30). Combination of Ctrl and Link using the IAF strategy provided a higher accuracy of 0.71 at a much earlier window (window 6). This model reached its peak performance at window 28 which was also more balanced in terms of class accuracy (see Figure 34). Ctrl+Bar had a high performance as well. However, it was at the last window and was not as balanced as Ctrl+Link (IAF).

5.1.3. Task Accuracy (Comprehension)

a. Training on each dataset alone and evaluating on Ctrl

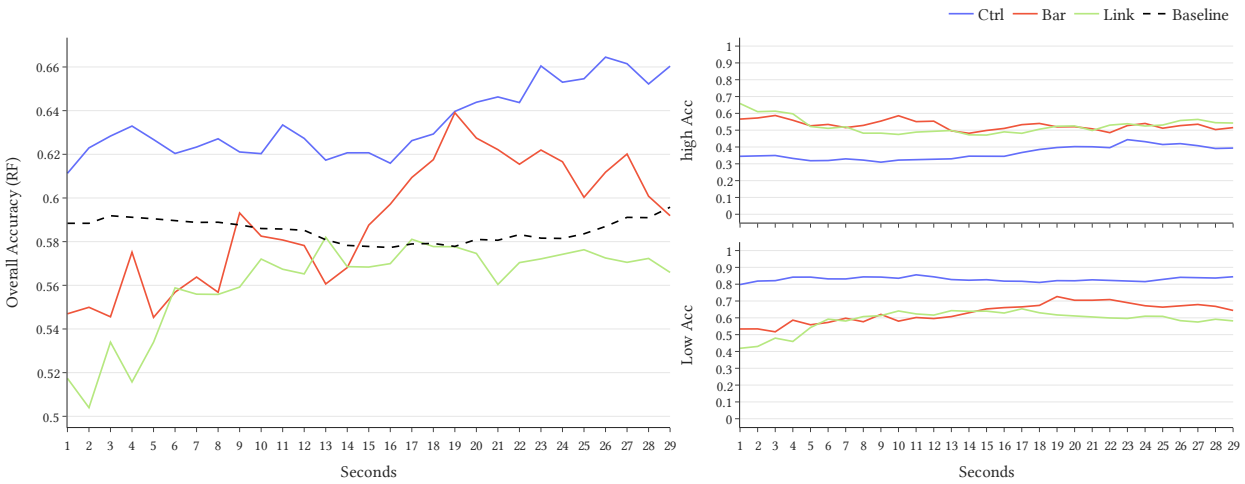

Figure 36: Results for Task Accuracy LR over different windows within-task when trained on each dataset alone

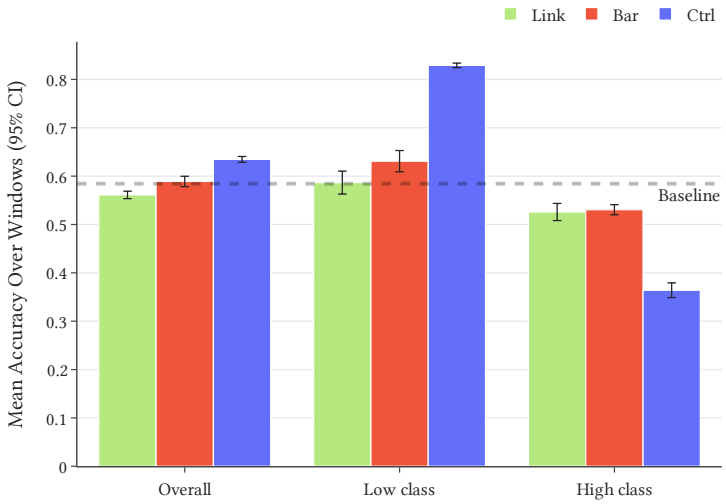

Figure 37: Average performance over all windows within-task for Task Accuracy LR when trained on each dataset alone

Table 28: Task Accuracy LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.59 (0.005)	-	-
Ctrl	0.63 (0.016)	0.83 (0.013)	0.36 (0.040)
Bar	0.59 (0.028)	0.63 (0.058)	0.53 (0.028)
Link	0.56 (0.020)	0.59 (0.062)	0.53 (0.047)

Figure 37 shows the performance of the LR classifier at predicting Task Accuracy when trained on each dataset alone. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl* > *Bar* = *Baseline* > *Link*. Both the Bar ($p < 0.0001$, $\eta^2 = 0.50$) and the Link dataset ($p < 0.0001$, $\eta^2 = 0.80$) yield lower performance than Ctrl. Ctrl has a fairly high accuracy for the low class which comes at the expense of misclassifying a large proportion of the high class.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

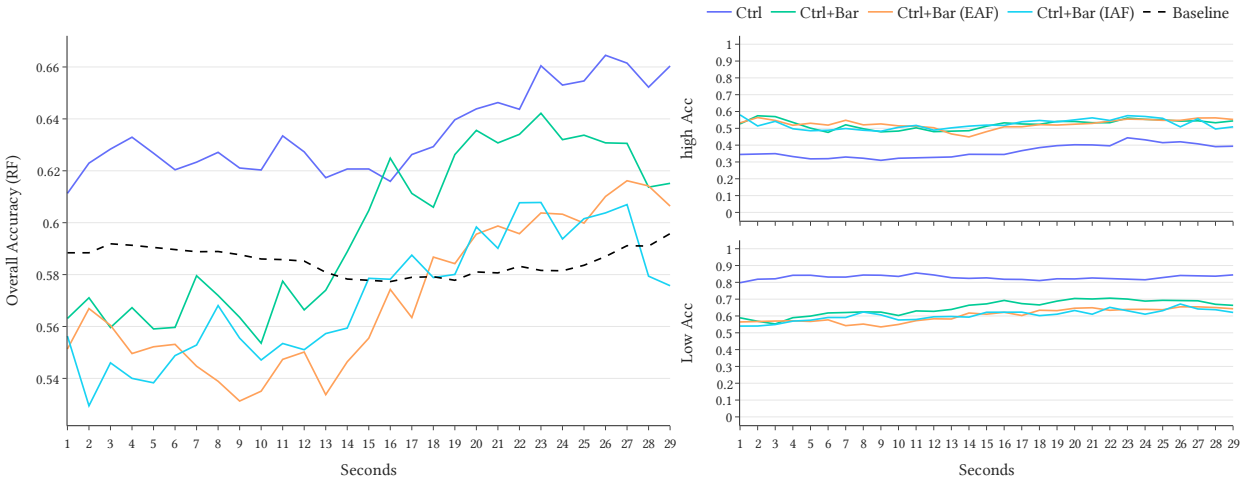

Figure 38: Results for Task Accuracy LR over different windows within-task when trained on the combination of Ctrl and Bar

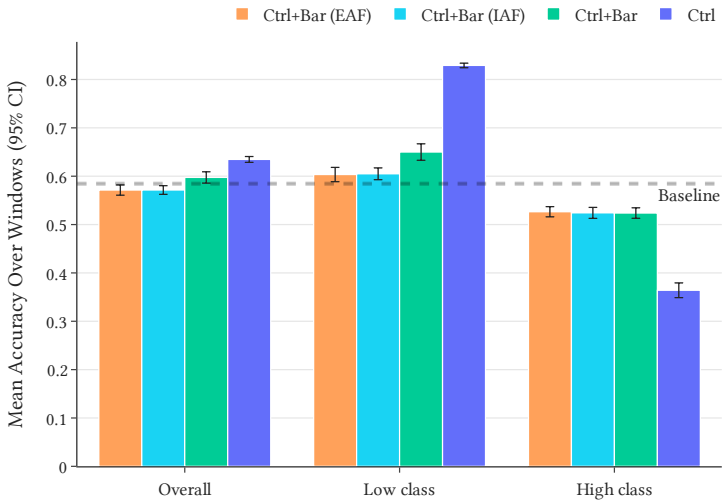

Figure 39: Average performance over all windows within-task for Task Accuracy LR when trained on the combination of Ctrl and Bar

Table 29: Task Accuracy LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.59 (0.005)	-	-
Ctrl	0.63 (0.016)	0.83 (0.013)	0.36 (0.040)
Ctrl+Bar	0.60 (0.031)	0.65 (0.045)	0.52 (0.028)
Ctrl+Bar (EAF)	0.57 (0.028)	0.60 (0.039)	0.53 (0.028)
Ctrl+Bar (IAF)	0.57 (0.024)	0.60 (0.032)	0.52 (0.030)

Figure 39 shows the performance of the LR classifier at predicting Task Accuracy when trained on the combination of Ctrl and Bar. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl* > *Ctrl+Bar* > *Baseline* > *Ctrl+Bar (IAF)* = *Ctrl+Bar (EAF)*. Combination of Ctrl and Bar (Ctrl+Bar) was not fruitful and resulted in a lower performance than Ctrl ($p < 0.0001$, $\eta^2 = 0.37$). Nevertheless, it is worth noting that Ctrl+Bar provided a better balance in terms of class accuracy than Ctrl. Neither of our strategies were able to make this combination worthwhile and both the EAF ($p = 0.002$, $\eta^2 = 0.17$) and the IAF ($p = 0.001$, $\eta^2 = 0.18$) yielded lower results than Ctrl+Bar. The results of EAF and IAF were quite similar ($p = 1$, $\eta^2 < 0.001$). This suggests that gaze features which were not significantly affected by the bar interventions had more relevance for predicting Task Accuracy than those which were significantly affected since excluding the affected ones had the same impact as imputing them.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

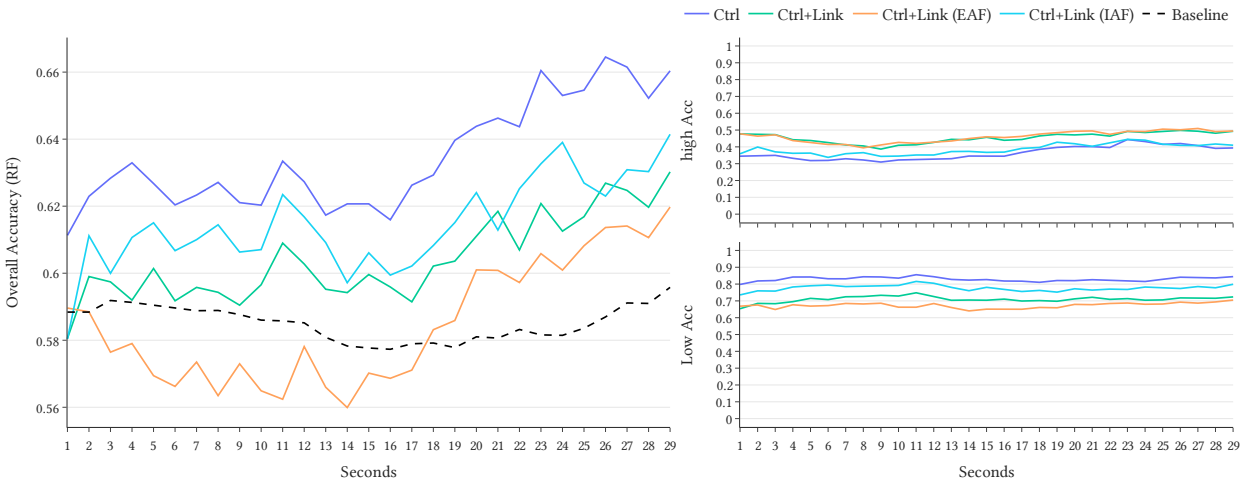

Figure 40: Results for Task Accuracy LR over different windows within-task when trained on the combination of Ctrl and Link

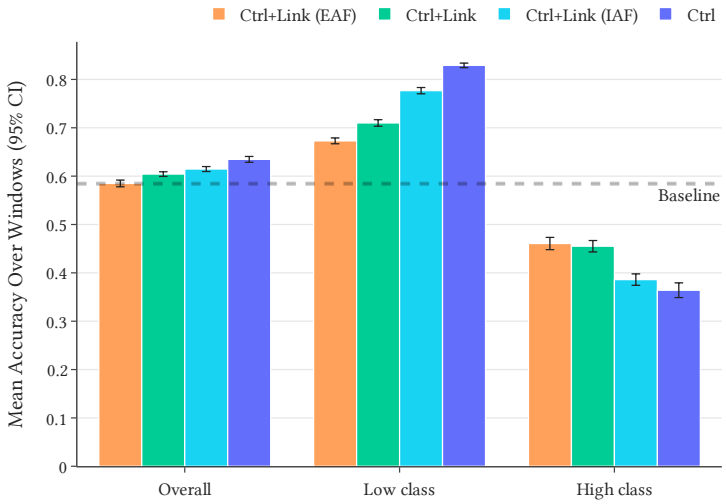

Figure 41: Average performance over all windows within-task for Task Accuracy LR when trained on the combination of Ctrl and Link

Table 30: Task Accuracy LR within-task results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.59 (0.005)	-	-
Ctrl	0.63 (0.016)	0.83 (0.013)	0.36 (0.040)
Ctrl+Link	0.60 (0.013)	0.71 (0.018)	0.46 (0.031)
Ctrl+Link (EAF)	0.58 (0.018)	0.67 (0.016)	0.46 (0.033)
Ctrl+Link (IAF)	0.61 (0.014)	0.78 (0.017)	0.39 (0.031)

Figure 41 shows the performance of the LR classifier at predicting Task Accuracy when trained on the combination of Ctrl and Link. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl > Ctrl+Link (IAF) > Ctrl+Link > Baseline = Ctrl+Link (EAF).* Combination of Ctrl and Link (Ctrl+Link) was not helpful and resulted in a lower performance than Ctrl ($p < 0.0001$, $\eta^2 = 0.53$). The EAF strategy made it even worse ($p < 0.0001$, $\eta^2 = 0.27$). The IAF strategy was more successful and resulted in a significant improvement ($p = 0.004$, $\eta^2 = 0.14$). However, this improvement was not large enough to beat Ctrl and was still significantly lower ($p < 0.0001$, $\eta^2 = 0.31$).

The peak performance of Ctrl was attained at window 26 with an overall accuracy of 0.66 (see Figure 36). 0.66 remained the best performance for this measure as combining neither of the adaptive datasets could offer any improvement.

5.2. Across Tasks Prediction Results

Similar to the previous section, we evaluate the performance of our strategies by running a one-way repeated measures ANOVA with accuracy as the dependent variable, strategy type as the factor, and time step (14) as the repeated measure.

5.2.1. VisLit

a. Training on each dataset alone and evaluating on Ctrl

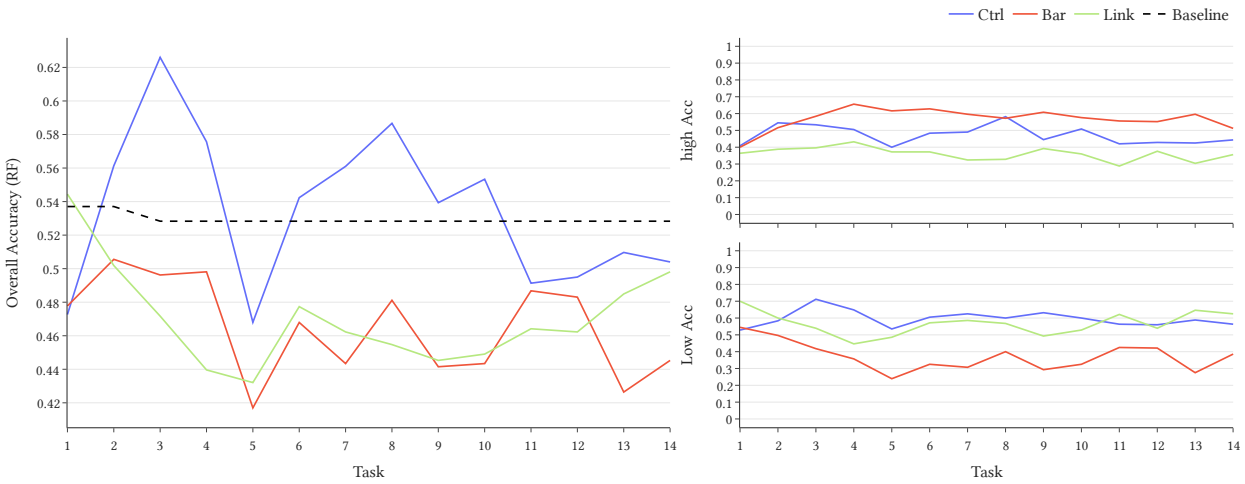

Figure 42: Results for VisLit RF over different windows across-tasks when trained on each dataset alone

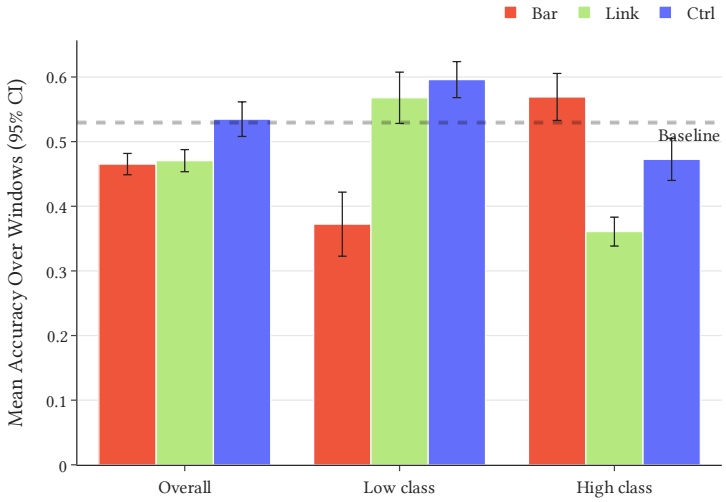

Figure 43: Average performance over all windows across-tasks for VisLit RF when trained on each dataset alone

Table 31: VisLit RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.003)	-	-
Ctrl	0.53 (0.045)	0.60 (0.047)	0.47 (0.055)
Bar	0.47 (0.029)	0.37 (0.086)	0.57 (0.063)
Link	0.47 (0.030)	0.57 (0.069)	0.36 (0.039)

Figure 43 shows the performance of the RF classifier at predicting VisLit when trained on each dataset alone across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl = Baseline > Link = Bar*. Similar to the within-task setup, both the Bar (p = 0.0001, η^2 = 0.45) and the Link dataset (p = 0.0003, η^2 = 0.41) yielded lower performance than Ctrl.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

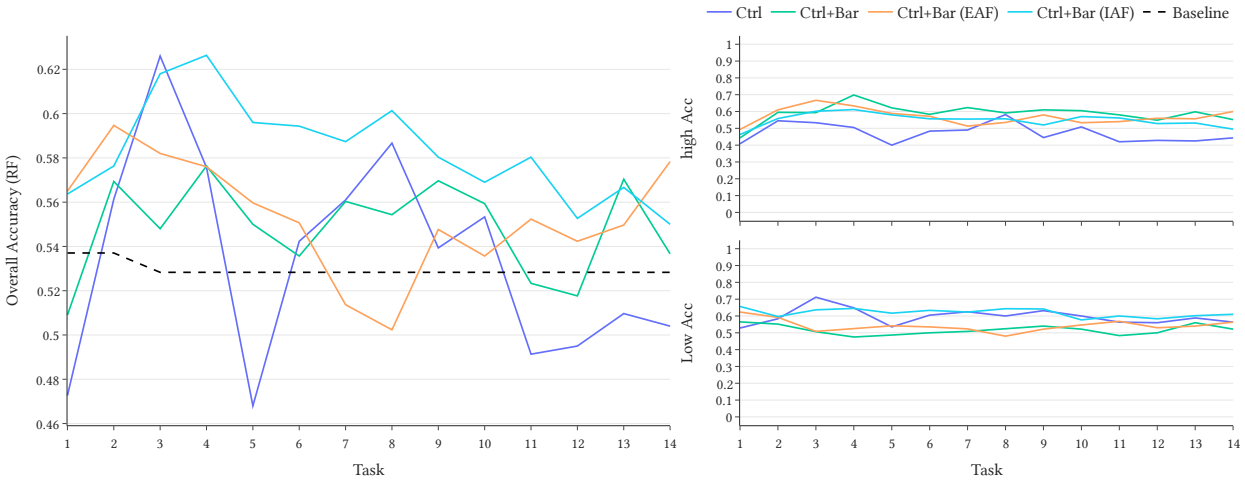

Figure 44: Results for VisLit RF over different windows across-tasks when trained on the combination of Ctrl and Bar

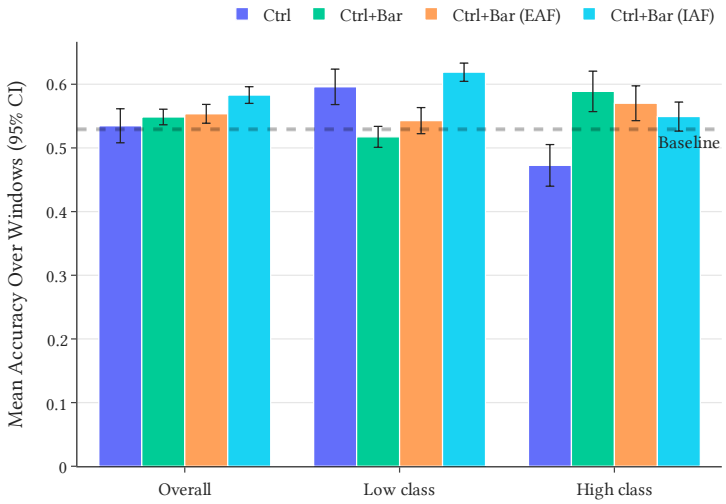

Figure 45: Average performance over all windows across-tasks for VisLit RF when trained on the combination of Ctrl and Bar

Table 32: VisLit RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.003)	-	-
Ctrl	0.53 (0.045)	0.60 (0.047)	0.47 (0.055)
Ctrl+Bar	0.55 (0.021)	0.52 (0.028)	0.59 (0.055)
Ctrl+Bar (EAF)	0.55 (0.026)	0.54 (0.036)	0.57 (0.047)
Ctrl+Bar (IAF)	0.58 (0.023)	0.62 (0.025)	0.55 (0.040)

Figure 45 shows the performance of the RF classifier at predicting VisLit when trained on the combination of Ctrl and Bar across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Bar (IAF) > Ctrl+Bar (EAF) = Ctrl+Bar = Ctrl = Baseline*. Unlike the within-task setup, combination of Ctrl and Bar (Ctrl+Bar) across-tasks did not result in a higher performance than Ctrl overall (p = 0.4, η^2 = 0.04). The EAF strategy reduced the gap in class performance, but it could not make this combination any more rewarding (p = 0.6, η^2 = 0.01). The IAF strategy, on the other hand, significantly improved performance overall (p = 0.002, η^2 = 0.38) with a higher accuracy for the low class. This strategy was also successful within-task.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

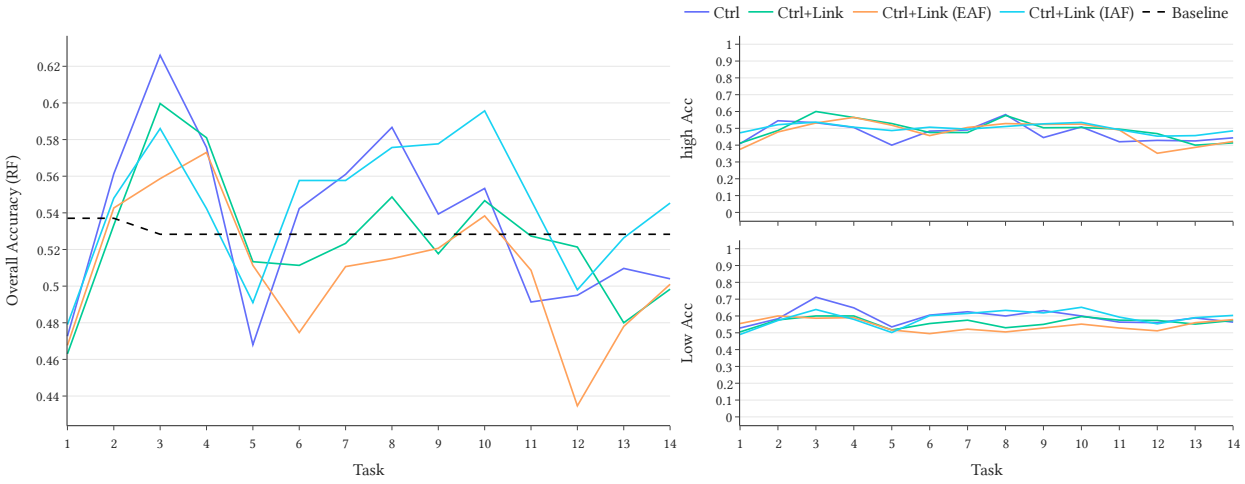

Figure 46: Results for VisLit RF over different windows across-tasks when trained on the combination of Ctrl and Link

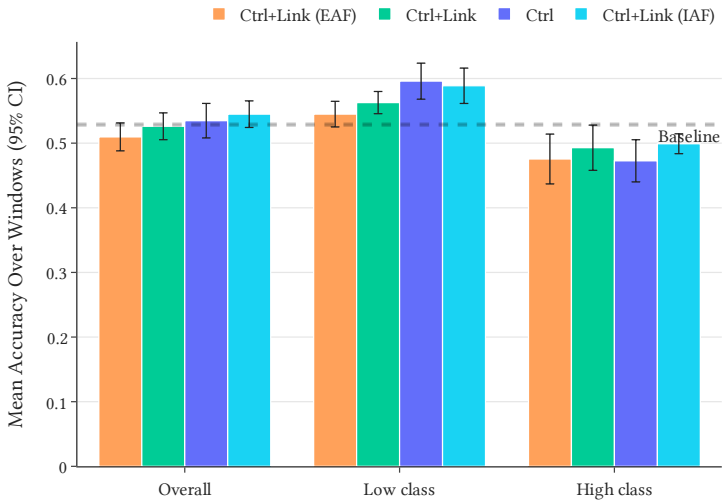

Figure 47: Average performance over all windows across-tasks for VisLit RF when trained on the combination of Ctrl and Link

Table 33: VisLit RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.003)	-	-
Ctrl	0.53 (0.045)	0.60 (0.047)	0.47 (0.055)
Ctrl+Link	0.53 (0.036)	0.56 (0.030)	0.49 (0.060)
Ctrl+Link (EAF)	0.51 (0.037)	0.54 (0.034)	0.48 (0.067)
Ctrl+Link (IAF)	0.54 (0.036)	0.59 (0.047)	0.50 (0.027)

Figure 41 shows the performance of the RF classifier at predicting VisLit when trained on the combination of Ctrl and Link across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link (IAF) = Ctrl = Baseline = Ctrl+Bar = Ctrl+Bar (EAF)*. Ctrl+Link, which was successful within-task, did not perform any better than Ctrl (p = 0.7, η^2 = 0.01) across-tasks. The EAF strategy was not helpful (p = 0.4, η^2 = 0.05) in either setup. The IAF strategy was more effective, but the improvement was not significant (p = 0.4, η^2 = 0.06). This strategy, was statistically similar to the naïve combination in the within-task setup, as well.

Table 34: VisLit RF Best performing windows (across-tasks)

Strategy	Window	Type	Overall Acc	Low Acc	High Acc
Ctrl	3 tasks	Window with highest overall acc	0.63	0.71	0.53
Ctrl+Bar (IAF)	4 tasks	Window with highest overall acc	0.63	0.65	0.61

Table 34 reports the accuracy of our best performing strategies at their viable window(s) at predicting VisLit across-tasks. The peak performance of Ctrl was attained at window 3 with an overall accuracy of 0.63 (see Figure 42). Combination of the Bar was more beneficial than Link. Ctrl+Bar (IAF) provided a similar accuracy to control a window later (window 4), but the performance was more balanced (see Figure 44). Although this improvement could be vluable, we were already able to achieve a peak accuracy of 0.66 with perfectly balanced class performance using the Ctrl+Link strategy after 12 seconds into the first task in the within-task setup which still remains our highest and earliest peak accuracy for VisLit.

5.2.2. ReadP

a. Training on each dataset alone and evaluating on Ctrl

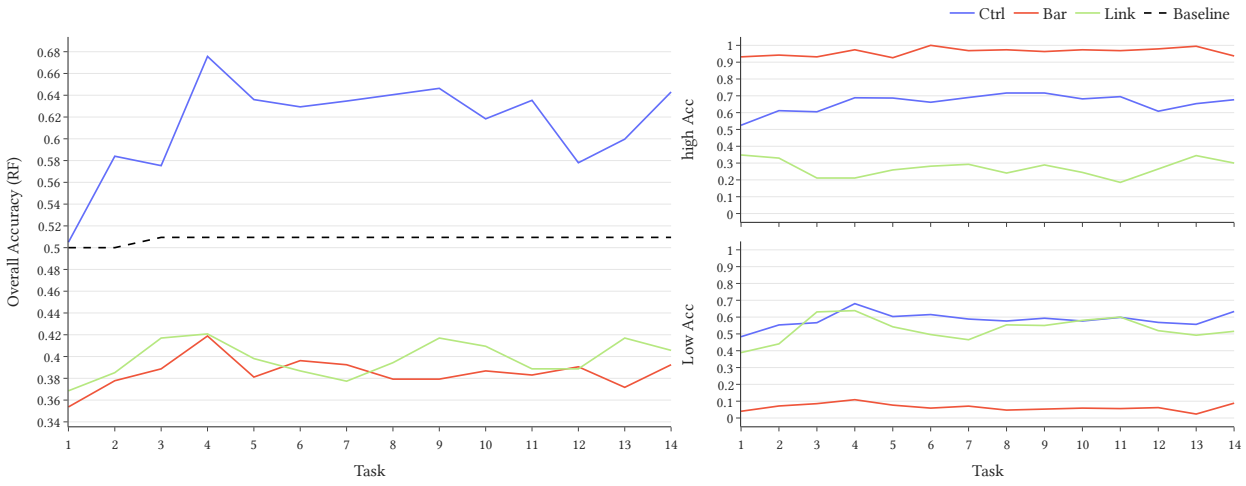

Figure 48: Results for ReadP RF over different windows across-tasks when trained on each dataset alone

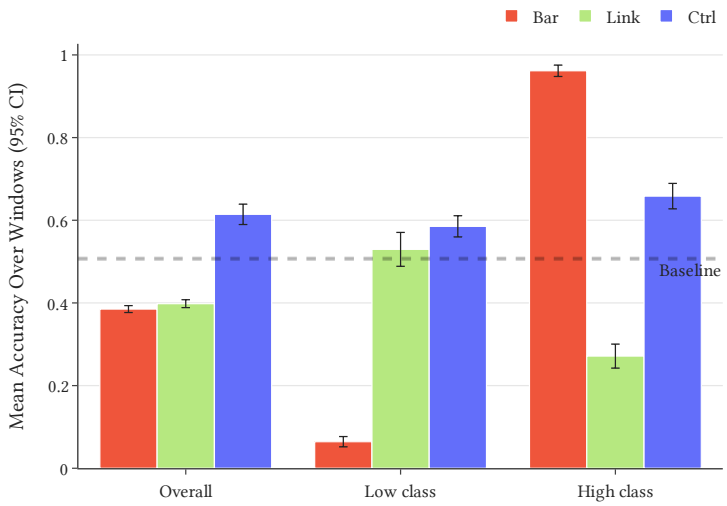

Figure 49: Average performance over all windows across-tasks for ReadP RF when trained on each dataset alone

Table 35: ReadP RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.51 (0.003)	-	-
Ctrl	0.62 (0.042)	0.59 (0.043)	0.66 (0.053)
Bar	0.39 (0.014)	0.06 (0.022)	0.96 (0.024)
Link	0.40 (0.017)	0.53 (0.071)	0.27 (0.050)

Figure 49 shows the performance of the RF classifier at predicting ReadP when trained on each dataset alone across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl* > *Baseline* > *Link* = *Bar*. Similar to the within-task setup, both the Bar (p < 0.0001, η^2 = 0.93) and the Link dataset (p < 0.0001, η^2 = 0.92) yielded lower performance than Ctrl. The Bar dataset had an exceptionally low performance on the low class which means that the model mistakenly predicted most of the low participants in the Ctrl dataset as high. Apparently, the model could not learn the behavior of the low ReadP users from the Bar dataset across tasks. This might suggest that the bar interventions mostly affected the gaze behavior of the low ReadP group over the entirety of the tasks.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

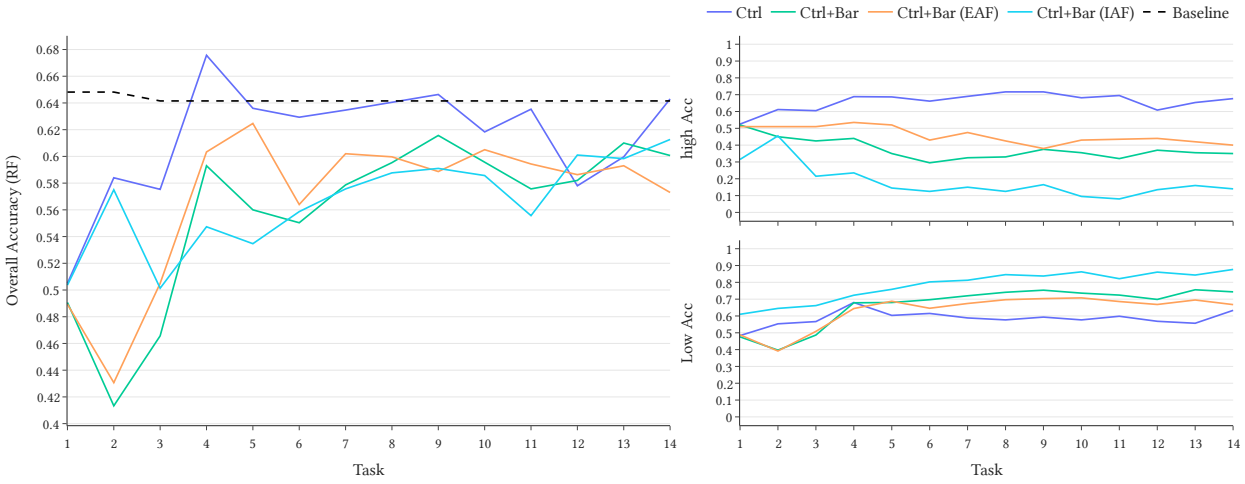

Figure 50: Results for ReadP RF over different windows across-tasks when trained on the combination of Ctrl and Bar

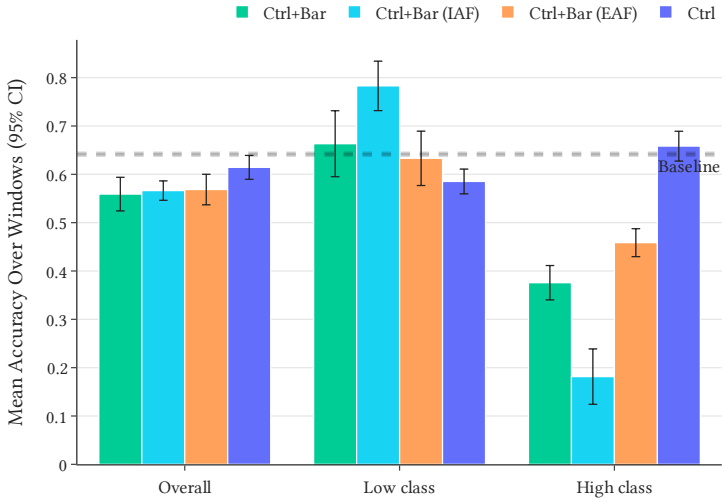

Figure 51: Average performance over all windows across-tasks for ReadP RF when trained on the combination of Ctrl and Bar

Table 36: ReadP RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.64 (0.002)	-	-
Ctrl	0.62 (0.042)	0.59 (0.043)	0.66 (0.053)
Ctrl+Bar	0.56 (0.060)	0.66 (0.118)	0.38 (0.062)
Ctrl+Bar (EAF)	0.57 (0.055)	0.63 (0.097)	0.46 (0.050)
Ctrl+Bar (IAF)	0.57 (0.035)	0.78 (0.089)	0.18 (0.099)

Figure 51 shows the performance of the RF classifier at predicting ReadP when trained on the combination of Ctrl and Bar across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Baseline* > *Ctrl* > *Ctrl+Bar (EAF)* = *Ctrl+Bar (IAF)* = *Ctrl+Bar*. Ctrl+Bar performed lower than Ctrl overall (p = 0.02, η^2 = 0.22). Neither the EAF (p = 0.8, η^2 = 0.007) nor the IAF strategy (p = 0.8, η^2 = 0.005) was able to provide a significant improvement. In the within-task setup, however, Ctrl+Bar and Ctrl+Bar (IAF) were both successful and yielded higher results than Ctrl.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

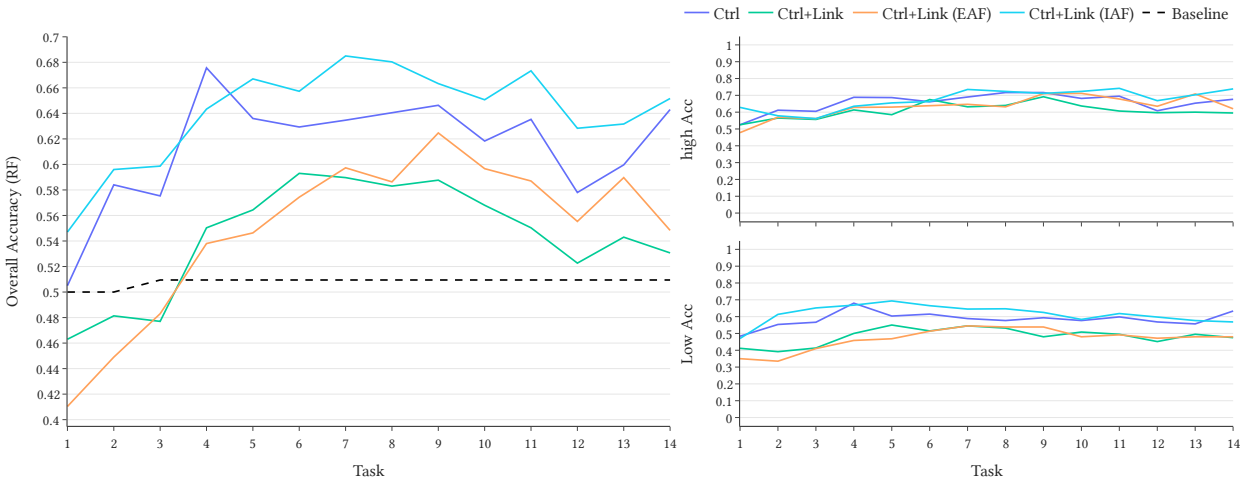

Figure 52: Results for ReadP RF over different windows across-tasks when trained on the combination of Ctrl and Link

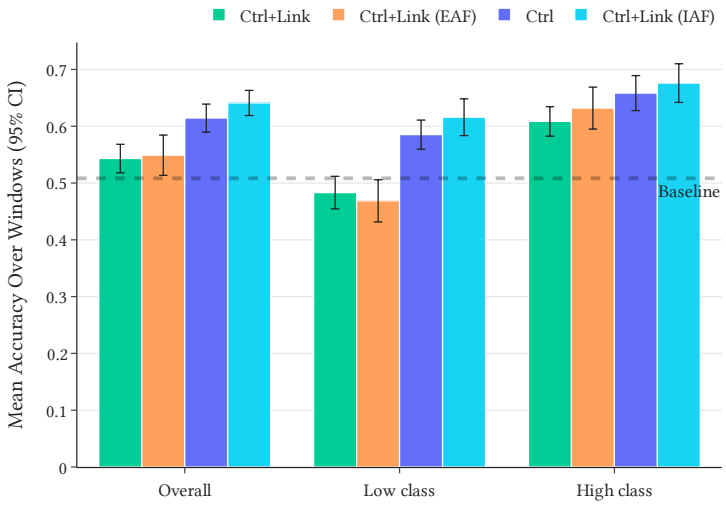

Figure 53: Average performance over all windows across-tasks for ReadP RF when trained on the combination of Ctrl and Link

Table 37: ReadP RF across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.51 (0.003)	-	-
Ctrl	0.62 (0.042)	0.59 (0.043)	0.66 (0.053)
Ctrl+Link	0.54 (0.043)	0.48 (0.050)	0.61 (0.045)
Ctrl+Link (EAF)	0.55 (0.062)	0.47 (0.064)	0.63 (0.064)
Ctrl+Link (IAF)	0.64 (0.038)	0.62 (0.056)	0.68 (0.059)

Figure 53 shows the performance of the RF classifier at predicting ReadP when trained on the combination of Ctrl and Link across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl+Link (IAF) = Ctrl > Ctrl+Bar (EAF) = Ctrl+Link > Baseline*. Ctrl+Link, despite being successful within-task, performed lower than Ctrl (p = 0.0004, η^2 = 0.41). The EAF strategy did not yield a significant improvement (p = 0.8, η^2 = 0.003). The IAF strategy, on the other hand, significantly improved performance (p < 0.0001, η^2 = 0.59). However, this improvement, as opposed to the within-task setup, was not large enough to outperform Ctrl (p = 0.1, η^2 = 0.1).

The peak performance of Ctrl was attained at window 4 with an overall accuracy of 0.68 (see Figure 48). Combination of the Link was more beneficial than the Bar. However, none of our strategies, across tasks, were able to perform better than Ctrl. The accuracy of 0.75, achieved within-task (task 1) using Ctrl+Link (IAF), remains our best result for this cognitive ability.

5.2.3. VerbalWM

a. Training on each dataset alone and evaluating on Ctrl

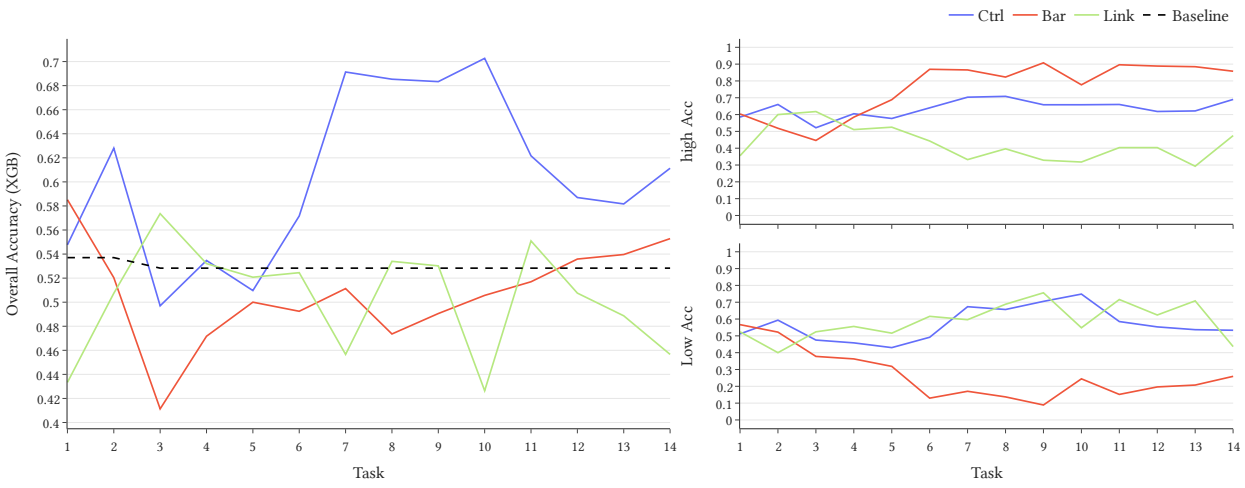

Figure 54: Results for VerbalWM XGB over different windows across-tasks when trained on each dataset alone

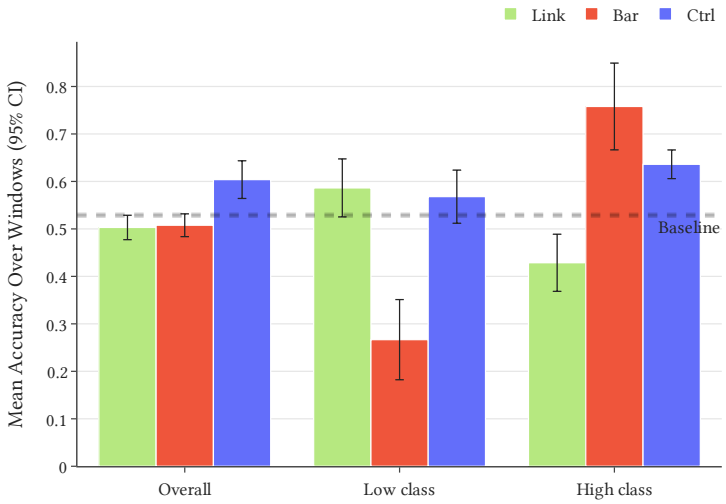

Figure 55: Average performance over all windows across-tasks for VerbalWM XGB when trained on each dataset alone

Table 38: VerbalWM XGB across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.003)	-	-
Ctrl	0.61 (0.067)	0.57 (0.093)	0.64 (0.053)
Bar	0.51 (0.042)	0.27 (0.146)	0.75 (0.158)
Link	0.50 (0.045)	0.59 (0.106)	0.43 (0.104)

Figure 55 shows the performance of the XGB classifier at predicting VerbalWM when trained on each dataset alone across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl* > *Baseline* = *Bar* = *Link*. Both the Bar (p = 0.0004, η^2 = 0.42) and the Link dataset (p = 0.0004, η^2 = 0.43) yielded lower performance than Ctrl. These datasets, however, performed higher than Ctrl within-task which was quite interesting.

b. Training on different combination strategies of Ctrl and Bar and evaluating on Ctrl

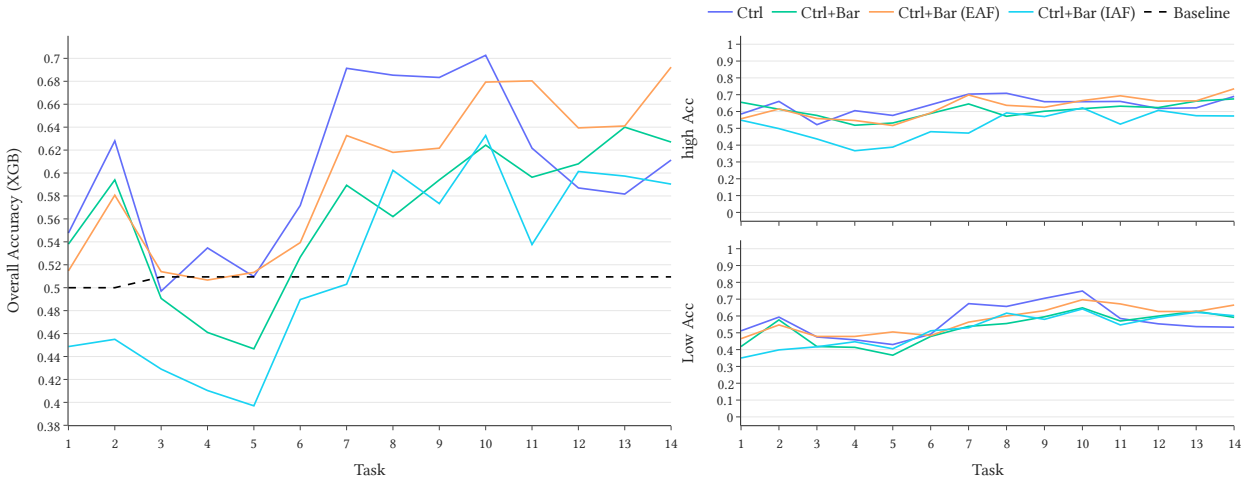

Figure 56: Results for VerbalWM XGB over different windows across-tasks when trained on the combination of Ctrl and Bar

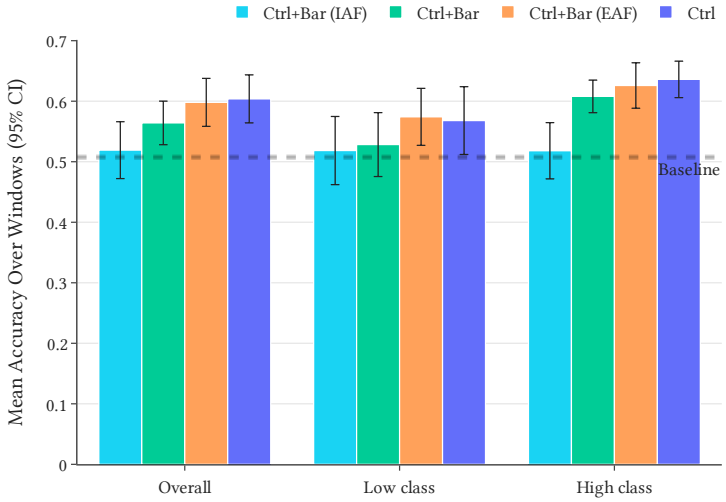

Figure 57: Average performance over all windows across-tasks for VerbalWM XGB when trained on the combination of Ctrl and Bar

Table 39: VerbalWM XGB across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.51 (0.003)	-	-
Ctrl	0.61 (0.067)	0.57 (0.093)	0.64 (0.053)
Ctrl+Bar	0.56 (0.062)	0.53 (0.091)	0.61 (0.047)
Ctrl+Bar (EAF)	0.60 (0.069)	0.57 (0.082)	0.63 (0.065)
Ctrl+Bar (IAF)	0.52 (0.081)	0.52 (0.098)	0.52 (0.081)

Figure 57 shows the performance of the XGB classifier at predicting VerbalWM when trained on the combination of Ctrl and Bar across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl = Ctrl+Bar (EAF) = Ctrl+Bar = Ctrl+Bar (IAF) = Baseline*. Ctrl+Bar performed lower than Ctrl (p = 0.2, η^2 = 0.08). This strategy, however, performed significantly better within-task owing to the higher performance of the Bar dataset on its own in that setup. None of our strategies were able to provide better results than Ctrl across-tasks.

c. Training on different combination strategies of Ctrl and Link and evaluating on Ctrl

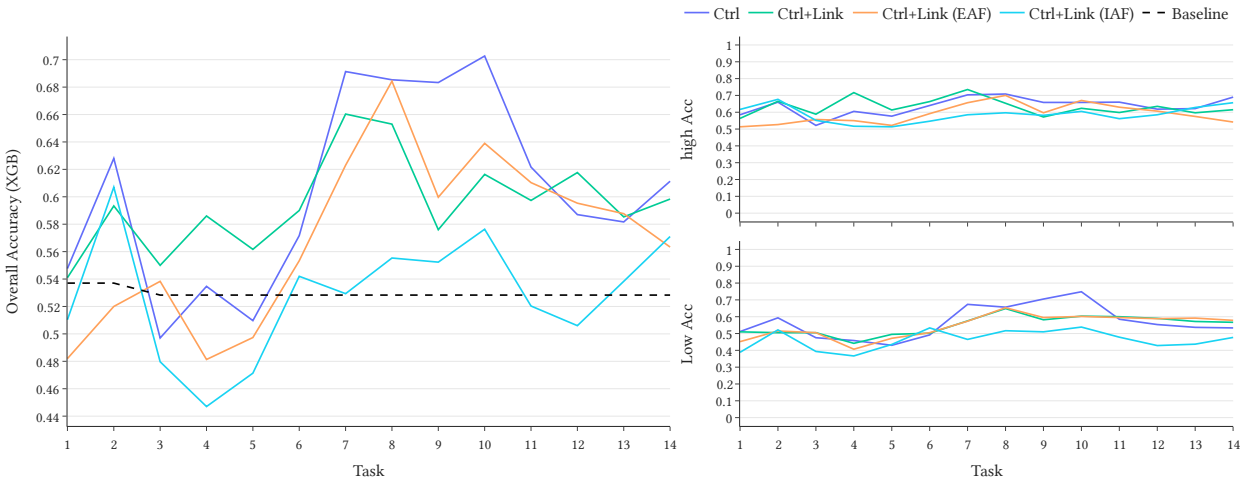

Figure 58: Results for VerbalWM XGB over different windows across-tasks when trained on the combination of Ctrl and Link

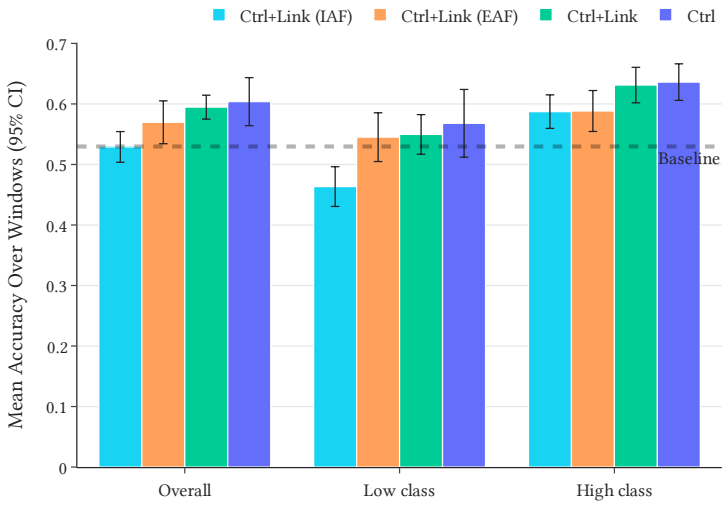

Figure 59: Average performance over all windows across-tasks for VerbalWM XGB when trained on the combination of Ctrl and Link

Table 40: VerbalWM XGB across-tasks results

Strategy	Overall Mean (SD)	Low Mean (SD)	High Mean (SD)
Baseline	0.53 (0.003)	-	-
Ctrl	0.61 (0.067)	0.57 (0.093)	0.64 (0.053)
Ctrl+Link	0.59 (0.034)	0.55 (0.057)	0.63 (0.051)
Ctrl+Link (EAF)	0.57 (0.061)	0.55 (0.070)	0.59 (0.059)
Ctrl+Link (IAF)	0.53 (0.044)	0.46 (0.057)	0.59 (0.048)

Figure 59 shows the performance of the XGB classifier at predicting VerbalWM when trained on the combination of Ctrl and Link across-tasks. We compare overall results by running post-hoc pairwise comparisons. The results of the comparisons show: *Ctrl = Ctrl+Link = Ctrl+Bar (EAF) > Baseline = Ctrl+Bar (IAF)*. Ctrl+Link had a similar performance to Ctrl (p = 0.7, η^2 = 0.007). This strategy, however, performed significantly better within-task owing to the higher performance of the Link dataset on its own in that setup. Similar to the combination of Bar and Ctrl, for the combination of Link and Ctrl, the EAF strategy was more successful than the IAF strategy, but they did not provide any improvement over Ctrl either.

The peak performance of Ctrl at predicting VerbalWM across tasks was reached at window 10 with an accuracy of 0.70 (see Figure 54). None of our combined datasets could provide a higher or even similar performance. Therefore, we do not report their peak accuracies.

6. Discussion

We evaluated the effectiveness of additional eye-tracking data, from two different adaptive user studies, in improving our accuracy for predicting three cognitive abilities (i.e., VisLit, ReadP, and Verbal WM), and two measures of task performance (i.e., Task Time and Task Accuracy). For this purpose, we trained the models proposed in [1] on the combined datasets and tested their performance on the control data, which represents users' normal gaze patterns prior to adaptation. Our results show that, overall, in the within-task predictions, training the models on the combined datasets significantly improved performance for all cognitive abilities and Task Time. Our evaluation of each of the adaptive datasets, when used alone, revealed that, in terms of VisLit, Link was more suitable to be combined than Bar. However, for ReadP and Task Time, Bar was more suitable. For Verbal WM, both adaptive datasets had similar performances and were both significantly better than Ctrl. This was a very interesting finding because training the model on either of the adaptive datasets, within-task, yielded a better performance than training it on the Ctrl dataset itself. It is possible that the interventions, provided in the adaptive studies, influenced users' gaze in a way that made it easier for the model to pick up on gaze patterns differentiating low vs high Verbal WM. Naïve combination of each adaptive dataset with Ctrl resulted in a significant improvement for all measures except Task Accuracy. Specifically, for Verbal WM, we were able to achieve significantly higher performance than the baseline. This was very encouraging as our previous performance for Verbal WM, within-task, was way below the baseline. Our IAF strategy, intended for making the combination more consistent, improved performance even further for Ctrl+Bar in terms of VisLit, and for Ctrl+Link in terms of ReadP. This strategy was most effective for cases in which the data to be combined was severely different and not as predictive as Ctrl.

According to our results, the combined datasets were more beneficial in the within-task setup than they were across-tasks. One possible explanation for this is that in the across-tasks predictions, for each participant, in each window, we build the feature vector by considering the entirety of the current task as well as all its preceding tasks. As a result, the impact of interventions would be more pronounced in each data point and build up as we accumulate more tasks. In the within-task predictions, however, we aggregate over the seconds elapsed within the same task. Therefore, given the finer granularity of the aggregation and the fact that interventions were not triggered at the first second, the data points would remain more similar to the control allowing for a more effective combination. Table 41 report the performance of our best strategies.

In addition to individually combining each adaptive dataset with control, we also tried combining all three datasets together and training our models on this larger dataset. However, doing so yielded no further improvement in performance. We suspect that the dominance of the adaptive data points over the control data, together with the increased disparity in the combination, prevented the model from properly learning the control gaze patterns.

Table 41: Best performing strategies and their best windows including within-task and across-tasks

Measure	Strategy	Best window	Overall Acc	Low Acc	High Acc
VisLit	Ctrl+Link	12 sec	0.66	0.66	0.66
ReadP	Ctrl+Link (IAF)	26 sec	0.75	0.76	0.74
VerbalWM	Ctrl	10 tasks	0.70	0.75	0.66
Task Time	Ctrl+Bar	29 sec	0.73	0.70	0.77
Task Accuracy	Ctrl	26 sec	0.66	0.84	0.42

Table 41 shows the performance of our best strategies at their best window. For VisLit, ReadP, and Task Time, we were able to achieve higher accuracies as a result of our combinations. For VerbalWM, as well, we were able to achieve higher results and outperform the previously unbeaten baseline within-task with a peak accuracy of 0.64. Although this accuracy was achieved after 29 seconds into the first task, the highest accuracy for this cognitive ability remains 0.70 which is reached after 10 tasks in the across-tasks prediction setup. None of our strategies in neither prediction setup could improve performance for task accuracy.

7. Conclusion

In this thesis, we extended a previous work on user modeling where eye-tracking was leveraged for predicting a user's levels of cognitive abilities and performance while processing magazine style narrative visualizations (MSNV), a common type of multimodal document which combines text and visualization. In particular, we improved the accuracy of these predictions by combining with the original eye-tracking data, which had been collected in a non-adaptive user study (Control), additional eye-tracking data from two other studies which used adaptations (Bar and Link) and training the proposed models on this larger dataset. The adaptive studies used the same MSNV documents as the non-adaptive control. However, they also provided users with gaze-driven adaptive support (i.e., by highlighting relevant parts of the visualization in the Bar study, and providing links between those and their corresponding references in the Link study) to facilitate processing of the MSNVs.

Our primary purpose for predicting users' cognitive abilities and performance is to regulate adaptation (i.e., by guiding what to adapt to, as well as when to adapt). Hence, we predict these measures based on users' normal gaze behavior prior to adaptation (control condition). In the adaptive studies, adaptive interventions impacted users' normal way of processing MSNVs. As a result, their gaze data, in the presence of adaptive support, did not represent their normal behavior. This inconsistency could prevent these data from being helpful. Nevertheless, not all aspects of their gaze processing were influenced by the interventions. We explored different combination strategies to mitigate the differences of adaptive gaze data from control so as to form a more consistent combination. Our results show that our IAF strategy, which imputed the affected eye-tracking features, in the adaptive datasets, with their control values, was able to maximize the gain and significantly improve performance for most of the measures especially in the within-task predictions.

Given the benefits of additional eye-tracking data in improving our classification accuracy, for future work, we encourage collection of more eye-tracking data in the control condition to further improve our results. In addition to task performance and task time, we also encourage prediction of other transient user states such as distraction, confusion, cognitive load, etc. which can further optimize our personalization mechanisms. Other directions for future work include conducting similar experiments with longer MSNV documents, which are more common in practice, to study how users behave in these situations where distraction or zoning out are also likely scenarios, and investigating consecutive adaptations where users' reaction to an adaptive intervention could be used as feedback to guide or rectify the subsequent one.

References

[1] Oswald Barral, Sébastien Lallé, Grigorii Guz, Alireza Iranpour, and Cristina Conati. 2020. Eye-Tracking to Predict User Cognitive Abilities and Performance for User-Adaptive Narrative Visualizations. In Proceedings of the 2020 International Conference on Multimodal Interaction (ICMI '20). Association for Computing Machinery, New York, NY, USA, 163–173. DOI:https://doi.org/10.1145/3382507.3418884

[2] S. Lallé, D. Toker and C. Conati, "Gaze-Driven Adaptive Interventions for Magazine-Style Narrative Visualizations," in IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 6, pp. 2941-2952, 1 June 2021, doi: 10.1109/TVCG.2019.2958540.

[3] S. Lallé, T. Wu and C. Conati, "Gaze-Driven Links for Magazine Style Narrative Visualizations," 2020 IEEE Visualization Conference (VIS), 2020, pp. 166-170, doi: 10.1109/VIS47514.2020.00040.

[4] Päivi Majaranta and Andreas Bulling. 2014. Eye tracking and eye-based human–computer interaction. In Advances in physiological computing. Springer, 39–65.

[5] Georg Buscher, Andreas Dengel, and Ludger van Elst. 2008. Eye movements as implicit relevance feedback. In CHI '08 Extended Abstracts on Human Factors in Computing Systems (CHI EA '08). Association for Computing Machinery, Florence, Italy, 2991–2996. DOI:https://doi.org/10.1145/1358628.1358796

[6] Tobias Appel, Natalia Sevcenko, Franz Wortha, Katerina Tsarava, Korbinian Moeller, Manuel Ninaus, Enkelejda Kasneci, and Peter Gerjets. 2019. Predicting Cognitive Load in an Emergency Simulation Based on Behavioral and Physiological Measures. In 2019 International Conference on Multimodal Interaction (ICMI '19). Association for Computing Machinery, Suzhou, China, 154–163. DOI:https://doi.org/10.1145/3340555.3353735

[7] Daria Bondareva, Cristina Conati, Reza Feyzi-Behnagh, Jason M. Harley, Roger Azevedo, and François Bouchet. 2013. Inferring Learning from Gaze Data during Interaction with an Environment to Support Self-Regulated Learning. In Proceedings of the 16th International Conference on Artificial Intelligence in Education. Springer, Memphis, TN, USA, 229–238.

[8] Shlomo Berkovsky, Ronnie Taib, Irena Koprinska, Eileen Wang, Yucheng Zeng, Jingjie Li, and Sabina Kleitman. 2019. Detecting Personality Traits Using Eye-Tracking Data. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, Glasgow, Scotland Uk, 1–12. DOI:https://doi.org/10.1145/3290605.3300451

[9] Paul Ayres and Gabriele Cierniak. 2012. Split-attention effect. In Encyclopedia of the Sciences of Learning. Springer, 3172–3175.

[10] Sidney D'Mello, Andrew Olney, Claire Williams, and Patrick Hays. 2012. Gaze tutor: A gaze-reactive intelligent tutoring system. Int. J. Hum.-Comput. Stud. 70, 5 (2012), 377–398. DOI:https://doi.org/10.1016/j.ijhcs.2012.01.004

[11] Florian Alt, Alireza Sahami Shirazi, Albrecht Schmidt, and Julian Mennenöh. 2012. Increasing the user's attention on the web: using implicit interaction based on gaze behavior to tailor content. In Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design. ACM, 544–553.

[12] Fabian Göbel, Peter Kiefer, Ioannis Giannopoulos, Andrew T. Duchowski, and Martin Raubal. 2018. Improving Map Reading with Gaze-adaptive Legends. In Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18). ACM, New York, NY, USA, 29:1–29:9. DOI:https://doi.org/10.1145/3204493.3204544

[13] Kenan Bektas, Arzu Cöltekin, Jens Krüger, and Andrew T. Duchowski. 2015. A testbed combining visual perception models for geographic gaze contingent displays. In Eurographics Conference on Visualization (EuroVis)-Short Papers. .
[14] Sébastien Lallé, Dereck Toker, and Cristina Conati. 2019. Gaze-Driven Adaptive Interventions for Magazine-Style Narrative Visualizations. IEEE Trans. Vis. Comput. Graph. (2019), 1–12. DOI:https://doi.org/10.1109/TVCG.2019.2958540
[15] Eli T. Brown, Alvitta Ottley, Hang Zhao, Quan Lin, Richard Souvenir, Alex Endert, and Ronald Chang. 2014. Finding waldo: Learning about users from their interactions. IEEE Trans. Vis. Comput. Graph. 20, 12 (2014), 1663–1672. DOI:https://doi.org/10.1109/tvcg.2014.2346575
[16] Ben Steichen, Cristina Conati, and Giuseppe Carenini. 2014. Inferring Visualization Task Properties, User Performance, and User Cognitive Abilities from Eye Gaze Data. ACM Trans. Interact. Intell. Syst. 4, 2 (2014), Article 11. DOI:https://doi.org/10.1145/2633043
[17] Scott A. Crossley, Stephen Skalicky, Mihai Dascalu, Danielle S. McNamara, and Kristopher Kyle. 2017. Predicting Text Comprehension, Processing, and Familiarity in Adult Readers: New Approaches to Readability Formulas. Discourse Process. 54, 5–6 (July 2017), 340–359. DOI:https://doi.org/10.1080/0163853X.2017.1296264
[18] Daria Bondareva, Cristina Conati, Reza Feyzi-Behnagh, Jason M. Harley, Roger Azevedo, and François Bouchet. 2013. Inferring Learning from Gaze Data during Interaction with an Environment to Support Self-Regulated Learning. In Proceedings of the 16th International Conference on Artificial Intelligence in Education. Springer, Memphis, TN, USA, 229–238.
[19] Samad Kardan and Cristina Conati. 2012. Exploring Gaze Data for Determining User Learning with an Interactive Simulation. In Proceedings on the International Conference on User Modeling, Adaptation, and Personalization (Lecture Notes in Computer Science). Springer Berlin Heidelberg, 126–138.
[20] Shahram Eivazi and Roman Bednarik. 2011. Predicting Problem-Solving Behavior and Performance Levels from Visual Attention Data. In Proceedings of the 2nd Workshop on Eye Gaze in Intelligent Human Machine Interaction, in conjunction with IUI 2011. ACM, Palo Alto, CA, USA, 9–16.
[21] Natasha Jaques, Cristina Conati, Jason M. Harley, and Roger Azevedo. 2014. Predicting Affect from Gaze Data during Interaction with an Intelligent Tutoring System. In Proceedings of the 12th International Conference on Intelligent Tutoring Systems. Springer, Honolulu, HI, USA, 29–38. DOI:https://doi.org/10.1007/978-3-319-07221-0_4
[22] Suowei Wu, Zhengyin Du, Weixin Li, Di Huang, and Yunhong Wang. 2019. Continuous Emotion Recognition in Videos by Fusing Facial Expression, Head Pose and Eye Gaze. In 2019 International Conference on Multimodal Interaction (ICMI '19). Association for Computing Machinery, Suzhou, China, 40–48. DOI:https://doi.org/10.1145/3340555.3353739
[23] Sidney D'Mello, Catlin Mills, Robert Bixler, and Nigel Bosch. 2017. Zone out no more: Mitigating mind wandering during computerized reading. In Proceedings of the 10th International Conference on Educational Data Mining. Wuhan, China, 8–15.
[24] Caitlin Mills, Robert Bixler, Xinyi Wang, and Sidney K. D'Mello. 2016. Automatic Gaze-Based Detection of Mind Wandering during Narrative Film Comprehension. International Educational Data Mining Society.
[25] Ronal Singh, Tim Miller, Joshua Newn, Eduardo Velloso, Frank Vetere, and Liz Sonenberg. 2020. Combining gaze and AI planning for online human intention recognition. Artif. Intell. 284, (July 2020), 103275. DOI:https://doi.org/10.1016/j.artint.2020.103275
[26] Stephen Akuma, Chrisina Jayne, Rahat Iqbal, and Faiyaz Doctor. 2015. Inferring Users' Interest on Web Documents Through Their Implicit Behaviour. In Proceedings of the International

Conference on Engineering Applications of Neural Networks. Springer, Rhodes, Greece, 315–324. DOI:https://doi.org/10.1007/978-3-319-23983-5_29

[27] Shamsi T. Iqbal, Piotr D. Adamczyk, Xianjun Sam Zheng, and Brian P. Bailey. 2005. Towards an index of opportunity: Understanding changes in mental workload during task execution. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. ACM, Portland, OR, USA, 311–320. DOI:https://doi.org/10.1145/1054972.1055016

[28] Sébastien Lallé, Cristina Conati, and Giuseppe Carenini. 2016. Predicting confusion in information visualization from eye tracking and interaction data. In Proceedings on the 25th International Joint Conference on Artificial Intelligence. AAAI Press, New York, NY, USA, 2529–2535.

[29] Nelson Silva, Tobias Schreck, Eduardo Veas, Vedran Sabol, Eva Eggeling, and Dieter W. Fellner. 2018. Leveraging eyegaze and time-series features to predict user interests and build a recommendation model for visual analysis. In Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications. ACM, Warsaw, Poland, 13:1-13:9.

[30] Sébastien Lallé, Cristina Conati, and Giuseppe Carenini. 2016. Prediction of individual learning curves across information visualizations. User Model. User-Adapt. Interact. 26, 4 (2016), 307–345. DOI:https://doi.org/10.1007/s11257-016-9179-5

[31] Cristina Conati, Sébastien Lallé, Md. Abed Rahman, and Dereck Toker. 2017. Further Results on Predicting Cognitive Abilities for Adaptive Visualizations. In Proceedings of the 26th International Joint Conference on Artificial Intelligence. AAAI Press, Melbourne, Australia, 1568–1574. DOI:https://doi.org/10.24963/ijcai.2017/217

[32] Matthew Gingerich and Cristina Conati. 2015. Constructing Models of User and Task Characteristics from Eye Gaze Data for User-Adaptive Information Highlighting. In Proceedings of the 29th Conference on Artificial Intelligence. AAAI Press, Austin, TX, USA, 1728– 1734.

[33] Ben Steichen, Giuseppe Carenini, and Cristina Conati. 2013. User-adaptive information visualization: using eye gaze data to infer visualization tasks and user cognitive abilities. In Proceedings of the 2013 international conference on Intelligent user interfaces (IUI '13). ACM, New York, NY, USA, 317–328. DOI:https://doi.org/10.1145/2449396.2449439

[34] Fabian Göbel, Peter Kiefer, Ioannis Giannopoulos, Andrew T. Duchowski, and Martin Raubal. 2018. Improving Map Reading with Gaze-adaptive Legends. In Proceedings of the 2018 ACM Symposium on Eye Tracking Research & Applications (ETRA '18). ACM, New York, NY, USA, 29:1– 29:9. DOI:https://doi.org/10.1145/3204493.3204544

[35] Zehui Zhan, Lei Zhang, Hu Mei, and Patrick S. W. Fong. 2016. Online Learners' Reading Ability Detection Based on Eye-Tracking Sensors. Sensors 16, 9 (September 2016), 1457. DOI:https://doi.org/10.3390/s16091457

[36] Thomas H. Carr. 1981. Building theories of reading ability: On the relation between individual differences in cognitive skills and reading comprehension. Cognition 9, 1 (January 1981), 73–114. DOI:https://doi.org/10.1016/0010- 0277(81)90015-9

[37] Shlomo Berkovsky, Ronnie Taib, Irena Koprinska, Eileen Wang, Yucheng Zeng, Jingjie Li, and Sabina Kleitman. 2019. Detecting Personality Traits Using Eye-Tracking Data. In Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, Glasgow, Scotland Uk, 1–12. DOI:https://doi.org/10.1145/3290605.3300451

[38] Heyjin Song and Nammee Moon. 2018. A Preference Based Recommendation System Design Through Eye-Tracking and Social Behavior Analysis. In Advances in Computer Science and Ubiquitous Computing (Lecture Notes in Electrical Engineering). Springer, Singapore, 1014–1019. DOI:https://doi.org/10.1007/978-981-10-7605-3_162

[39] Dereck Toker, Cristina Conati, and Giuseppe Carenini. 2019. Gaze analysis of user characteristics in magazine style narrative visualizations. User Model. User-Adapt. Interact. to appear, (2019).
[40] J. Boy, R.A. Rensink, E. Bertini, and J.-D. Fekete. 2014. A Principled Way of Assessing Visualization Literacy. IEEE Trans. Vis. Comput. Graph. 20, 12 (December 2014), 1963– 1972. DOI:https://doi.org/10.1109/TVCG.2014.2346984
[41] Paul Meara. 2010. EFL Vocabulary Tests (second edition ed.). Lognostics, Swansea: Wales.
[42] C. Conati, G. Carenini, E. Hoque, B. Steichen, and D. Toker, "Evaluating the impact of user characteristics and different layouts on an interactive visualization for decision making," in Computer Graphics Forum, 2014, vol. 33, pp. 371–380, doi: https://doi.org/10.1111/cgf.12393.
[43] Sébastien Lallé, Cristina Conati, and Roger Azevedo. 2018. Prediction of Student Achievement Goals and Emotion Valence during Interaction with Pedagogical Agents. In Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems. IFAAMAS, Stockholm, Sweden, 1222–1231.
[44] Ramkumar Rajendran, Anurag Kumar, Kelly E. Carter, Daniel T. Levin, and Gautam Biswas. 2018. Predicting Learning by Analyzing Eye-Gaze Data of Reading Behavior. International Educational Data Mining Society.
[45] Lucia Mason, Patrik Pluchino, Maria Caterina Tornatora, and Nicola Ariasi. 2013. An Eye-Tracking Study of Learning From Science Text With Concrete and Abstract Illustrations. J. Exp. Educ. 81, 3 (July 2013), 356–384. DOI:https://doi.org/10.1080/00220973.2012.727885
[46] Lucia Mason, Maria Caterina Tornatora, and Patrik Pluchino. 2013. Do fourth graders integrate text and picture in processing and learning from an illustrated science text? Evidence from eye-movement patterns. Comput. Educ. 60, 1 (January 2013), 95–109. DOI:https://doi.org/10.1016/j.compedu.2012.07.011
[47] Yoav Benjamini and Yosef Hochberg. 1995. Controlling the false discovery rate: a practical and powerful approach to multiple testing. J. R. Stat. Soc. Ser. B Methodol. 57, 1 (1995), 289–300.
[48] M. Kuhn. 2008. Building predictive models in R using the caret package. J. Stat. Softw. 28, 5 (2008), 1–26. DOI:http://dx.doi.org/10.18637/jss.v028.i05
[49] Pedregosa F, Varoquaux, Ga"el, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. Journal of machine learning research. 2011;12(Oct):2825–30.
[50] Marilyn L Turner and Randall W Engle. 1989. Is working memory capacity task dependent? J. Mem. Lang. 28, 2 (April 1989), 127–154. DOI:https://doi.org/10.1016/0749-596X(89)90040-5
[51] Leana Copeland, Tom Gedeon, and Sabrina Caldwell. 2015. Effects of text difficulty and readers on predicting reading comprehension from eye movements. In 2015 6th IEEE International Conference on Cognitive Infocommunications (CogInfoCom).
[52] Sidney D'Mello, Jonathan Cobian, and Matthew Hunter. 2013. Automatic Gaze-Based Detection of Mind Wandering during Reading. In Proceedings of the 6th International Conference on Educational Data Mining. Memphis, TN, USA, 364–365.
[53] Young-Min Jang, Rammohan Mallipeddi, and Minho Lee. 2014. Identification of human implicit visual search intention based on eye movement and pupillary analysis. User Model. User-Adapt. Interact. 24, 4 (October 2014), 315–344. DOI:https://doi.org/10.1007/s11257-013-9142-7
[54] https://www.cs.ubc.ca/~lalles/MetroQuest/study_website/OP.php

Appendices

Appendix A: Reproducing control results within-task

Red dashed lines show previous results from [1] computed using the Caret package in R. The blue solid lines show the same results reproduced using the scikit-learn package in Python.

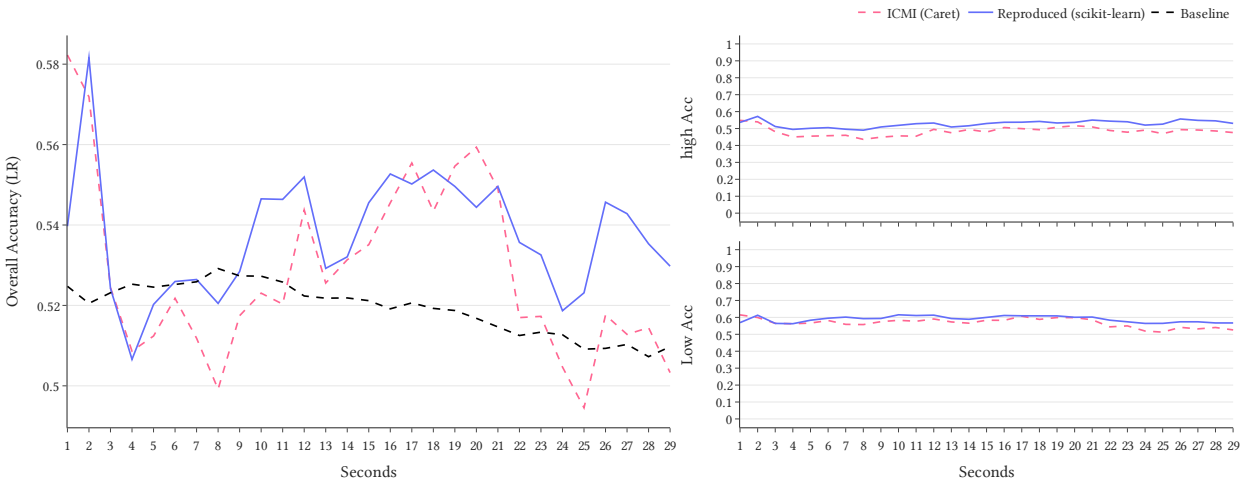

Figure 60: VisLit LR within-task results (Caret vs scikit-learn)

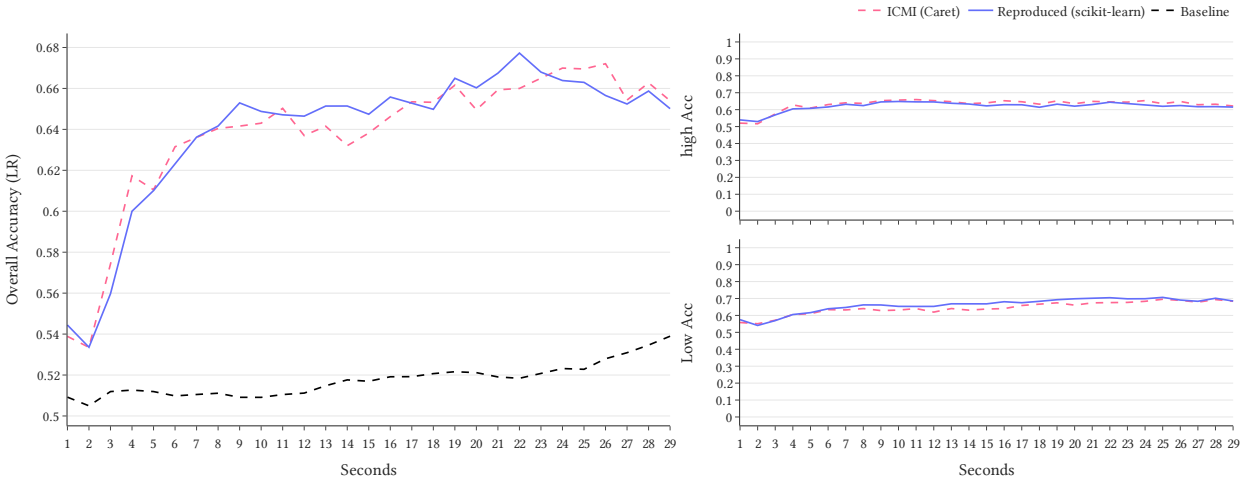

Figure 61: ReadP LR within-task results (Caret vs scikit-learn)

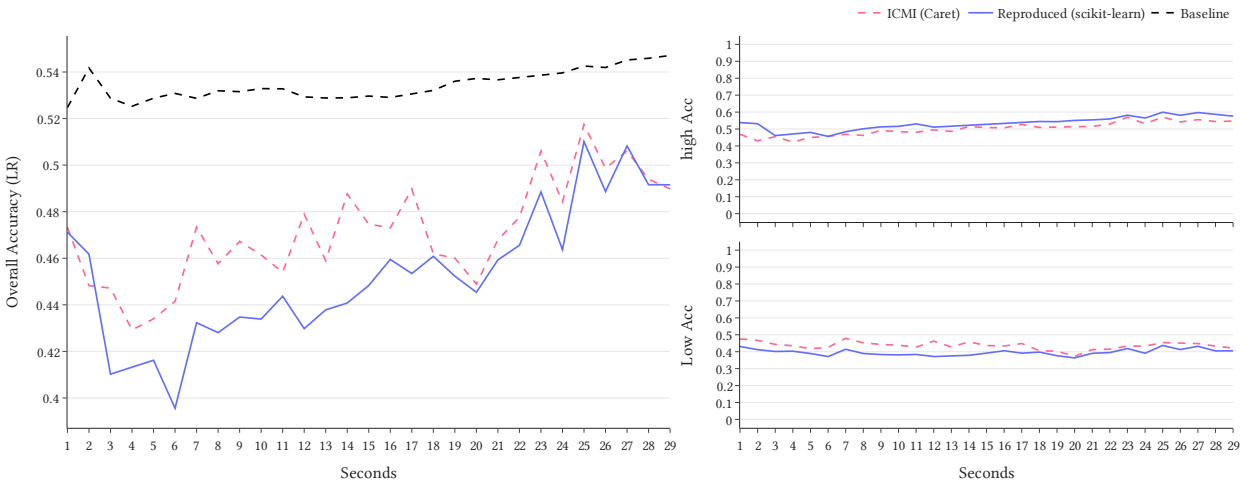

Figure 62: VerbalWM LR within-task results (Caret vs scikit-learn)

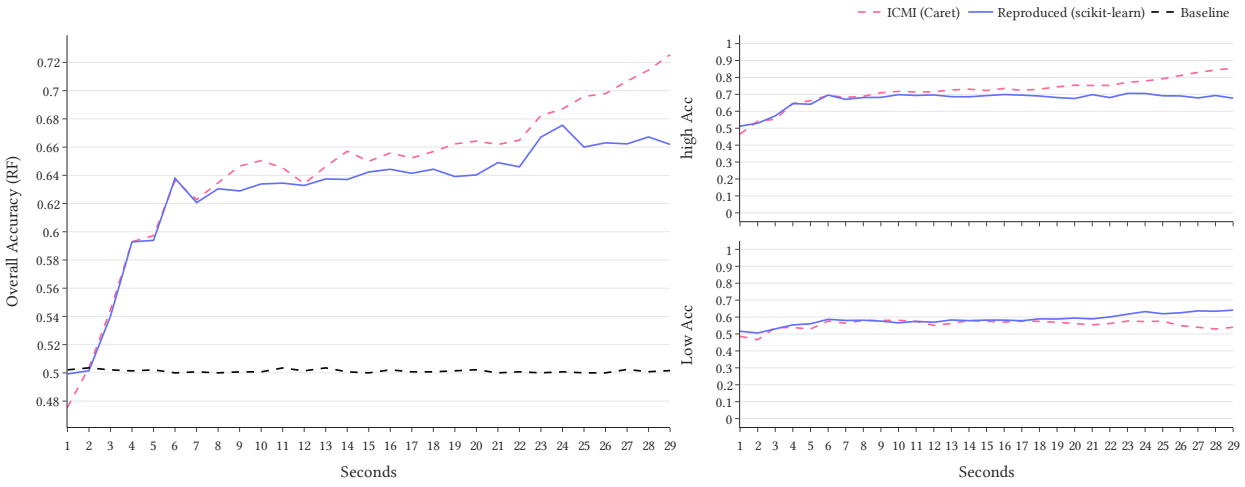

Figure 63: Task-Time RF within-task results (Caret vs scikit-learn)

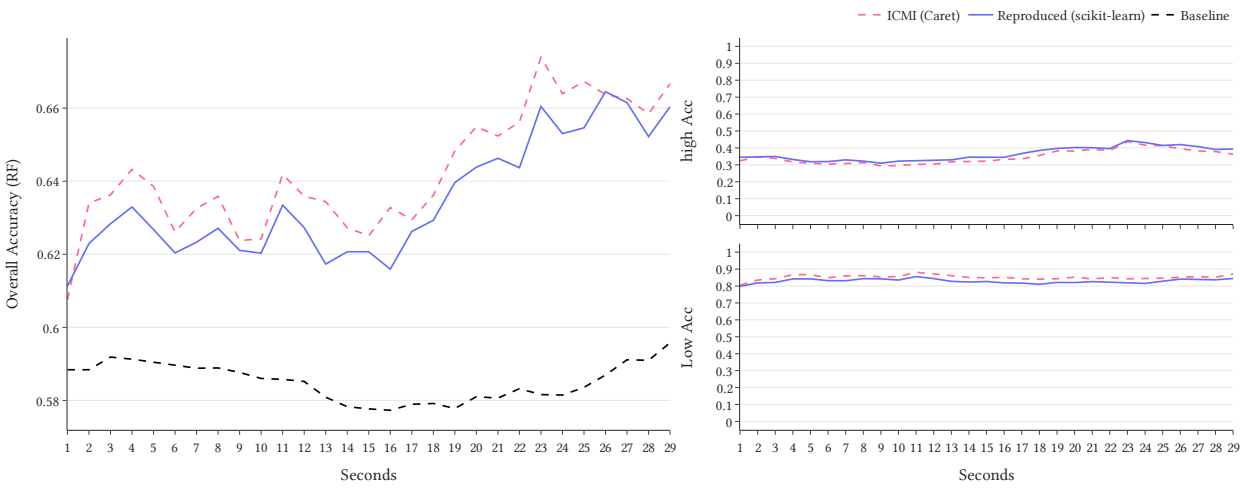

Figure 64: Task-Accuracy RF within-task results (Caret vs scikit-learn)

Appendix B: Reproducing control results across-tasks

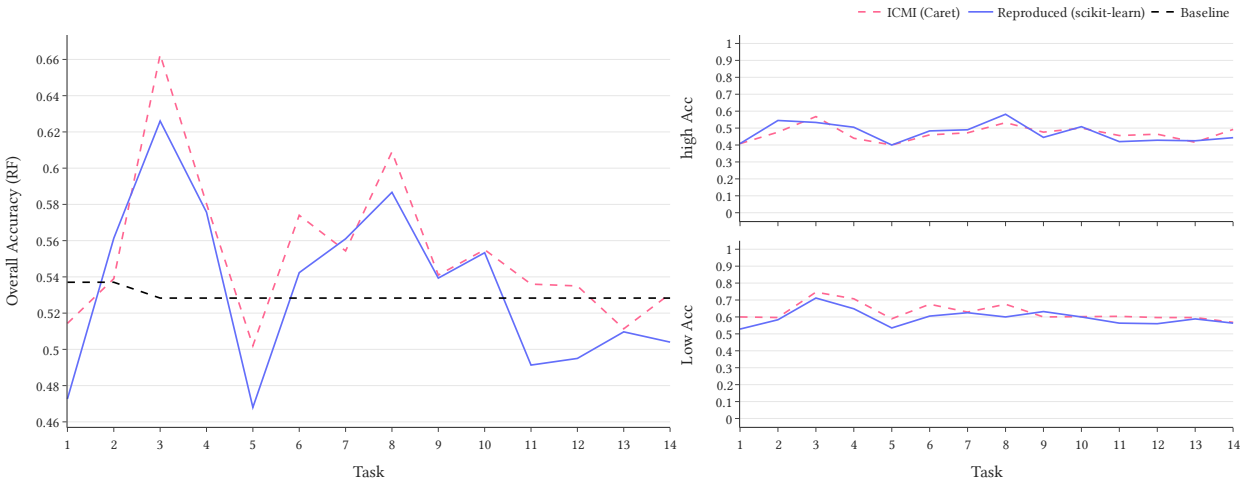

Figure 65: VisLit RF across-tasks results (Caret vs scikit-learn)

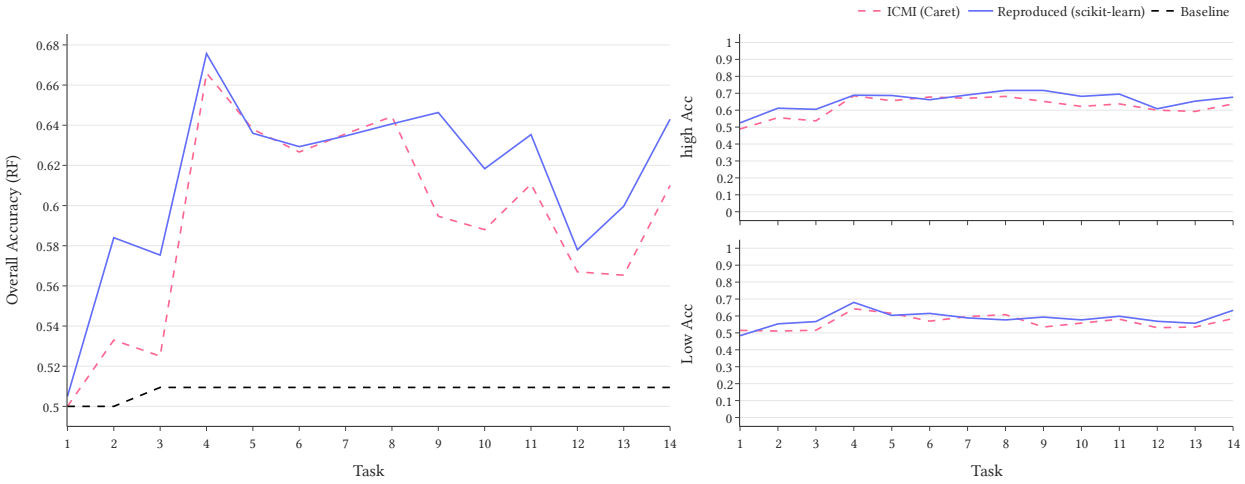

Figure 66: ReadP RF across-tasks results (Caret vs scikit-learn)

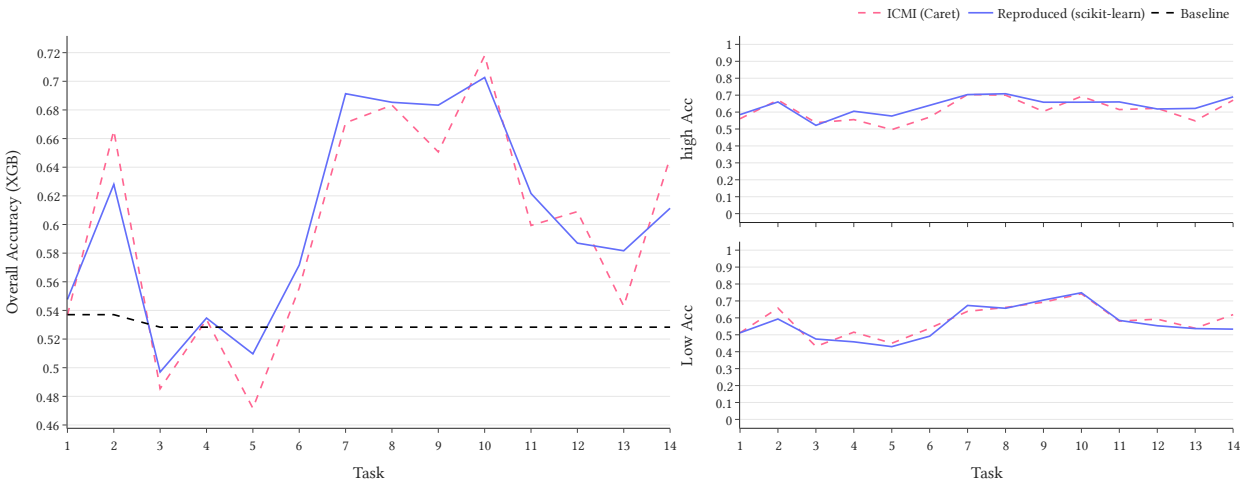

Figure 67: VerbalWM XGB across-tasks results (Caret vs scikit-learn)