INSTITUTE
OF PHONETICS
Faculty of Arts
Charles University

BACHELOR THESIS

Prokop Hanžl

Implementation of Active Learning in Perception Experiments

Využití metody aktivního učení v percepčních experimentech

Prague 2025

Supervisor: Ing. Tomáš Bořil, Ph.D.

I would like to thank Dr. Tomáš Bořil for his support and endless patience throughout the writing of my thesis. Furthermore, I would like to extend my gratitude to my parents and sister, who have been nothing but supportive, and my friends for lifting me up when I felt like I was losing one battle after another against Python and its evil libraries.

I declare that the following thesis is my own work for which I used only the sources and literature mentioned, and that this thesis has not been used in the course of other university studies or in order to acquire the same or another type of diploma.

Prohlašuji, že jsem práci vypracoval samostatně, že jsem řádně citoval všechny použité prameny a literaturu a že práce nebyla využita v rámci jiného vysokoškolského studia či k získání jiného nebo stejného titulu.

V Praze dne 15. 8. 2025 *Prokop Hanžl*

Abstract

This thesis introduces AsTRiQue, a Python-based framework for applying active machine learning (AL) to two-alternative forced-choice perception experiments in phonetics. The system aims to model participant responses efficiently, reducing the number of required labeled trials without compromising accuracy. Performance was assessed using pre-existing data on adult perception of Czech children's sibilants using virtual agents. Results from experimental simulations indicate that AsTRiQue can meaningfully reduce participant workload while maintaining high predictive performance. Although currently limited to binary classification with logistic regression, the open source nature of the system allows integration of alternative learner models and extension to more complex paradigms. The findings suggest that AL can enable richer, more varied stimulus sets while keeping experiments manageable, offering a promising direction for enhancing efficiency and flexibility in phonetic research.

Keywords: phonetics, perception experiment, active learning, machine learning

Abstrakt

Tato práce představuje systém AsTRiQue vytvořený v jazyce Python pro využití metody aktivního strojového učení (AL) v percepčních experimentech ve fonetice ve schématu nuceného výběru ze dvou možností. Cílem systému je efektivně modelovat odpovědi respondentek a respondentů a tím snížit počet stimulů, které musí ohodnotit, bez výrazného snížení kvality dat. Systém byl otestován na již existujících datech o percepci sykavek v řeči českých dětí dospělými posluchači za pomoci virtuálních agentů. Výsledky simulací ukazují, že AsTRiQue dokáže snížit zátěž respondentek a respondentů za zachování vysoké přesnosti výsledků. Ačkoli je systém v současnosti omezen na binární klasifikaci za využití logistické regrese, lze integrovat jiné modely učení a rozšířit jej na složitější experimentální paradigmata; jeho zdrojový kód je open source. Výsledky naznačují, že AL dokáže umožnit užití bohatších a pestřejších sad stimulů v percepčních experimentech bez jejich prodloužení. AL tedy představuje perspektivní směr pro zvýšení efektivity a flexibility fonetického výzkumu.

Klíčová slova: fonetika, percepční experiment, aktivní učení, strojové učení

Contents

1 Introduction

1.1 Perception Experiments

Perception experiments, where participants are presented with various stimuli and asked to answer questions about them, are an integral part of phonetics research. They allow researchers to investigate how speech sounds are categorized, differentiated, or evaluated by real speakers.

1.1.1 Types of Perception Experiments

There are various paradigms for perception experiments, which McGuire (2021) groups into three main categories: *discrimination*, *identification*, and *scaling* (though there is some overlap between these groups).

Discrimination tasks investigate speakers' abilities to distinguish several stimuli from each other. As such, multiple stimuli are often presented in each trial. This group includes various specific experimental paradigms. In *same-different* (AX) tasks, two stimuli, either identical or differing in some given aspect, are played with a brief pause between them (*interstimulus interval*, ISI), and the participant is asked to judge whether they were the same or different. In *matching-to-sample* (ABX, AXB, XAB) tasks, three stimuli are presented in order, and the participant makes a judgment whether stimulus X is identical or closest to stimulus A or B. *Two-alternative forced choice* (2AFC) experiments can be understood in two ways: the name can either refer to the number of options a participant is forced to choose from in each trial (for example, judging whether a sentence played is declarative or interrogative with no other possibilities presented; some AX, ABX or even identification experiments may fall into this category), or it can be used for experiments where a participant makes a choice between two stimuli

based on a given criterion or set of criteria (which is distinct from AX and ABX experiments); for this thesis, the former definition will be of relevance. Other such designs include *4-interval forced choice* (4IAX) and *category change* (oddball) tasks.

In identification tasks, participants assign explicit labels to the stimuli presented. *Labeling* (forced-choice identification, classification) tasks require participants to assign each stimulus to a class, often from a set or predetermined options (e.g., "did you hear /s/ or /ʃ/?"), or sometimes without this limitation (e.g., "write what you heard"). A simple (yet powerful) identification task is the *yes-no* experiment: in it, the participant makes a yes-no judgment about each stimulus based on a given criterion. Another subtype of identification tasks is the *oddity* experiment, in which one of the stimuli presented in each trial is somehow different, and the participant judges which one is the "odd one out".

Scaling tasks (also known as rating tasks) broadly refer to all kinds of experiments where the participant is asked to evaluate the stimuli they hear on a given scale (which can be visual or numeric).

Of course, these paradigms may be combined within a single experimental session. However, as McGuire (2021) suggests, they may affect each other in undesirable ways if proper care is not taken to ensure the validity of the experiment.

1.1.2 Limitations in Perception Testing

Perception experiments can be constrained by practical factors; notably, the number of stimuli that can be included in each participant's trial set is limited. Past experiments in psychology with various tasks have shown that participant attention peaks at the start, then steadily drops off, especially after the first 30 minutes (Vallesi et al., 2021; Reteig et al., 2019; Mackworth, 1948).

Therefore, researchers must strike a balance between presenting enough stimuli to each participant to obtain meaningful and generalizable results, and limiting the total number of trials to maintain participant engagement and data quality.

If too few stimuli are used, important patterns in the data may be missed; if too many are used, participants may become fatigued, leading to lower-quality responses. This trade-off poses a central challenge for experimental design in phonetics.

Common ways of approaching this issue include increasing the number of participants (e.g., Erickson et al., 2023; Li and Mok, 2023; Manker, 2020), or including breaks throughout the testing session (e.g., Cooper and Cooper, 2023; Hanžlová and Bořil, 2023; Lung et al., 2023). Less commonly, testing can be split into multiple sessions on different days (e.g., Pisoni, 1973). Clearly, all of these solutions come with their drawbacks; they increase the time required to run the experiment, and therefore the researcher's (and, in some cases, each participant's) workload.

A novel way of combating this problem is proposed by Einfeldt et al. (2024): modeling the participant's behavior live and serving them only the most informative stimuli using an active machine learning paradigm.

1.2 Machine Learning

Machine learning (ML) can be broadly defined as a subfield of artificial intelligence concerned with developing and studying algorithms or computer systems that improve their performance at a given task with experience. In the words of Mitchell (1997, p. 2):

> A computer program is said to **learn** from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Every machine learning program must, by this definition, have a clear definition of E, T, and P.

He further illustrates the definition on the example of a computer program learning to play checkers. The performance P (measured as win rate) of an

algorithm that learns to play this game may improve at its class of tasks T (playing checkers) with experience E, which it obtains by playing games against itself.

1.2.1 Types of Machine Learning

Nowadays, there is a vast array of tools at a ML engineer's disposal, each suited to particular types of data and problem structures. Four main types discussed by Jo (2021) are *supervised*, *unsupervised*, *semi-supervised* and *reinforcement learning*.

As described by Murphy (2012), supervised learning is a type of machine learning in which a system is trained on a dataset that includes both inputs and their corresponding correct outputs. Its goal is to find a mathematical function that accurately maps inputs to outputs, so that it can predict a correct output for any input, including new ones it has not seen in its training data. For example, in a phonetics setting, a supervised learning model might be used to predict a speaker's age (output) based on various acoustic measurements of a recording of their voice (input), after having been trained on many instances where the age of the speaker is known. When the outputs are categorical (e.g., speaker gender, provided each speaker can be assigned to one of a predefined set of genders), the task is called *classification* or *pattern recognition*. When the outputs are continuous (e.g., speaker age in years), it is known as *regression*.

Supervised learning algorithms are plentiful. *Logistic regression* is used to examine the relationship between a set of explanatory variables (*predictors* or *covariates*) and a binary outcome variable (Hosmer et al., 2013, p. 1), such as predicting whether an email is spam (1) or not (0). *Support vector machines* (SVMs), originally developed for binary classification, but extendable to regression and multi-class classification (Murphy, 2012, p. 497), work by mapping input data into a high-dimensional space and finding a linear boundary (a hyperplane) that maximizes the margin between two classes (Cortes and Vapnik, 1995). Unlike in logistic regression, only a subset of the data points (the *support vectors*, which

lend the algorithm its name) determines this boundary. *Neural networks* are computational models inspired by the structure of the human brain, consisting of many simple processing units (neurons) interconnected in a layered structure (with an input layer, output layer, and optional hidden layers in between). They acquire knowledge through a learning process that adjusts the strengths of individual connections (synaptic weights) between the neurons, enabling complex tasks such as pattern recognition. A significantly different architecture from logistic regression and SVMs, neural networks are able to better adapt to their environment and store experiential knowledge to improve their performance over time (Haykin, 2009, pp. 1–2). Other supervised learning algorithms include the k-nearest neighbors (KNN) algorithm, naive Bayes classification, and decision trees (Jo, 2021, p. 11).

In unsupervised learning, only unlabeled data is provided and the algorithm is tasked with finding regularities or patterns worth investigating in it. This problem, sometimes called *knowledge discovery*, is a vastly more general one than in supervised learning, and arguably bears more resemblance to human learning (Murphy, 2012, p. 2, 9). One common such algorithm described by Murphy (2012, p. 352) is k-means clustering, which partitions a dataset into a fixed number of groups (*clusters*), based on their similarity. Each point is assigned to the nearest cluster center, and each center is updated as the average of its assigned points. This process repeats until assignments stabilize. Another is the agglomerative hierarchial clustering (AHC) algorithm. In it, a hierarchy of clusters is built by repeatedly merging the two closest clusters based on a chosen measure of dissimilarity. Its result is a tree-like structure (*dendrogram*) showing how clusters are formed (Hastie et al., 2009, pp. 520–522). A relatively recent addition to the unsupervised learning arsenal are generative adversarial networks (GANs), as proposed by Goodfellow et al. (2014). In this paradigm, two models are trained at once: a *generator*, which attempts to produce data similar to real examples, and a *discriminator*, which tries to tell the difference between real and generated data. The generator improves as it learns to fool the discriminator, while the

discriminator improves by getting better at detecting fakes. Through this game of cat and mouse, the generator learns to create increasingly realistic data.

Sometimes, *self-supervised learning* algorithms, where the model automatically generates supervisory signals derived directly from the input data, are included under unsupervised learning (X. Liu et al., 2021).

Semi-supervised learning stands at the meeting point of supervised and unsupervised learning. According to Jo (2021, pp. 14–15), it combines a small number of labeled examples with a large number of unlabeled ones to improve performance in tasks like classification and regression. It uses the labeled data to guide learning and the unlabeled data to capture patterns or groupings in the input. One example mentioned is the combination of naive Bayes and expectation-maximization (EM) algorithms.

In reinforcement learning, as described by Jo (2021, pp. 15–16), an algorithm learns by interacting with its environment. It receives inputs (called *percepts*), takes actions in response, and gets feedback in the form of rewards or penalties. Over time, the algorithm adjusts its behavior to maximize rewards and avoid penalties, gradually making discoveries about which actions are most beneficial in different situations.

1.2.2 Active Machine Learning

Active learning (AL) is a form of supervised machine learning in which the learning algorithm is not trained on a given fully labeled dataset, but rather actively chooses the data it is trained on (Settles, 2009, p. 4). The central insight behind AL is that not all data points are equally informative for the learning task. For instance, in a binary classification setting, examples that lie near the model's decision boundary are typically more ambiguous and thus will carry more weight and deliver faster improvements to the model compared to data points further away from it. By strategically selecting the most informative data points during training, an AL algorithm's aim is to reach high prediction accuracy with significantly fewer

labeled examples (Settles, 2012, p. 5), regardless of the underlying classifier's architecture.

A typical AL setup involves two components: a *learner* and an *oracle* (Settles, 2009, p. 5). The learner is the model being trained, which has access to a large pool of unlabeled data. With each iteration, it selects one or more data points from the unlabeled pool that it deems most informative, based on a query strategy such as uncertainty sampling (S. Liu and Li, 2023; Guo and Wang, 2015), query-by-committee (R. Wang et al., 2012; Seung et al., 1992), or expected model change maximization (Cai et al., 2017, 2013). The oracle (often a human annotator or a simulated "virtual agent" in experimental settings) then provides the correct labels for the queried instances. These newly labeled data points are added to the training set, and the model is retrained. This loop continues until a stopping criterion is met, such as reaching a performance threshold or exhausting the labeling budget (for more detailed examples of various proposed stopping criteria, see **bloodgood_method_2014**, Olsson and Tomanek, 2009, or Vlachos, 2008).

The various AL approaches share a common principle: they assume that the learner algorithm keeps an internal measure of stability or confidence, and that the benefits of active learning diminish once this measure plateaus or begins to decline (Settles, 2012, p. 44).

1.2.3 AL in Perception Experiments

In the context of a perception experiment, the participant plays the role of the oracle, answering the learner algorithm's queries. The algorithm selects the stimuli that will be presented to the participant, who in turn makes judgments about them just as they would in a regular perception experiment.

The goal is to model the participants' response behavior efficiently, essentially creating a "digital version" of each one separately. By querying each participant's classifications of the stimuli carefully selected by the learner algorithm, the system can learn to predict how they would likely classify the rest. Once the

model is confident enough in its predictions, the experiment is stopped, reducing participant workload. At this point in the experiment, the system's predictions for the remaining stimuli not heard by the participant are accepted as equivalent to the participant's would-be judgments.

This approach, first explored by Einfeldt et al. (2024), will be the focus of this thesis. In it, I will present the base of a Python framework for applying AL in 2AFC perception experiments and running tests using data from previously conducted perception experiments.

2 Method

2.1 System Overview

To implement the AL paradigm described in the previous chapter, I developed a framework in Python 3.13.3 (van Rossum, 1995), named **AsTRiQue**; short for *Ask the Right Questions*, to reflect the essence of what it aims to do. It operates in two distinct modes. In the first, it is used with pre-existing, fully labeled datasets to simulate participant behavior via virtual agents. In this setup, all responses of a given participant are known in advance, and the system selectively queries a subset of these responses in order to model that participant's classification behavior and predict their likely judgments on the remaining items. The second mode supports live interaction with a human participant, where responses are collected in real time only for the stimuli chosen by the system. In both modes, the aim is to identify the smallest and most informative set of participant judgments required to build an accurate predictive model of their perceptual decision-making. Information about the technical implementation, alongside code snippets, is provided in Section 2.2.1.

2.1.1 Tools and Libraries

AsTRiQue was developed and currently runs inside a Jupyter notebook environment (Kluyver et al., 2016). Data processing is handled by NumPy 2.2.5 (Harris et al., 2020) and Pandas 2.2.3 (The Pandas Development Team, 2020; McKinney, 2010). The logistic regression model logic is processed by scikit-learn 1.6.1 (Pedregosa et al., 2011). The model evaluation charts in the software and this thesis are created using Matplotlib 3.10.3 (The Matplotlib Development Team, 2025;

Hunter, 2007). Statistical analyses were conducted using SciPy 1.15.2 (Virtanen et al., 2020).

2.1.2 Configuration

The framework has a set of parameters that must be configured before it can be used. A brief description for each setting is listed in Table 2.1.

Key	**Description**
`DATA_PATH`	Path to the CSV table containing information about the stimuli.
`FILENAME_COL`	Name of the column containing the file names of the stimuli.
`PREDICTOR1`	Name of the column containing values for the first numeric predictor.
`PREDICTOR2`	Name of the column containing values for the second numeric predictor.
`LABEL_MAPPING`	Mapping of binary output (0 or 1) to text labels (stimulus classes).
`STRATIFIED_SAMPLING_RESOLUTION`	Resolution of the 2D square grid to sample from.
`MODEL_CERTAINTY_CUTOFF`	Main stopping criterion (end when all model prediction certainties exceed this value).
`MIN_ITERATIONS`	Minimum number of iterations before the stopping criterion is considered.
`CLEANSER_FREQUENCY`	Insert a high-certainty sample every n iterations.
`PROCESSED_PATH`	Path where logs for the current runtime will be saved.
`PARTICIPANT_CSV_DIR`	Path to the directory containing participant lookup tables for virtual agent use.
`TARGET`	Name of the column in the participant lookup tables with the correct classification for each stimulus.
`PARTICIPANT_TO_MODEL`	ID of the participant currently being modeled as a virtual agent.

Table 2.1: An overview of AsTRiQue's configuration

On input, AsTRiQue accepts a table containing metadata about the speech stimuli used in the experiment. For the purposes of the active learning algorithm itself, the actual acoustic files are not required. The path to the table containing the required metadata is specified via the `DATA_PATH` parameter. In the virtual

agent mode, this metadata alone (specifically the two numeric predictors and the corresponding target classifications) is sufficient for running the system. In the live participant mode, the acoustic stimuli are, of course, necessary to present to the listener, but even here the algorithm operates solely on the associated predictor values and participant responses, without directly processing the audio signal. Within the metadata table, the `FILENAME_COL` parameter identifies the column containing the file names or paths for individual stimuli, while the parameters `PREDICTOR1` and `PREDICTOR2` define the names of the columns containing the values of the two numeric predictors used during model training.

As the system works within a binary classification paradigm, outputs must be mapped to two distinct class labels. While these are internally encoded as 0 and 1, the `LABEL_MAPPING` parameter allows the researcher to specify corresponding human-readable class names (e.g., "s" and "z").

The initial sample selection is governed by a grid-based stratified sampling procedure, the granularity of which is defined by `STRATIFIED_SAMPLING_RESOLUTION` (stratified sampling resolution). This ensures that the two-dimensional predictor space is covered uniformly. Further details on this process are provided in Section 2.1.3.

The initial sample selection is governed by a grid-based stratified sampling procedure, the granularity of which is defined by the stratified sampling resolution parameter. This parameter is specified as a single integer n, which determines the resolution of a square grid of size $n \times n$ overlaid on the two-dimensional predictor space. The predictor values form the axes of this space, and the grid divides it into equally sized cells. From each grid cell containing at least one stimulus, a single stimulus is randomly selected to ensure uniform coverage of the predictor space. Further details on this process are provided in Section 2.1.3.

The stopping condition is controlled by the `MODEL_CERTAINTY_CUTOFF` (model certainty cutoff) parameter, a number between 0 (any predictions are accepted; runtime stops after the initial stratified/random sampling phase discussed in

Section 2.1.3) and 1 (only fully unambiguously certain predictions are accepted; the participant must label every single stimulus) which specifies a threshold for the model's confidence in its predictions; stimuli labeled by the participant do not have prediction certainty values and their participant classifications are accepted as correct. Once the model prediction certainty values for all unlabeled samples exceed this value, the runtime is eligible to conclude (more information regarding the calculation of this metric is provided in Section 2.2.1). However, to prevent premature termination caused by potentially unreliable model estimates (such as those resulting from lucky, or, indeed, unlucky random selections of training items that appear too easily classifiable) the parameter `MIN_ITERATIONS` (minimum iterations) sets a minimum number of iterations that must be completed before the stopping criterion can be evaluated. This ensures the model has sufficient data to provide a robust prediction before the experiment ends.

To mitigate participant fatigue during data collection, the system allows for periodic insertion of high-certainty "cleanser" items. The frequency of these insertions is set by the `CLEANSER_FREQUENCY` parameter. This design feature is further discussed in Section 2.1.3.

During execution, the system generates a log of the runtime with participant answers and predictions, which are saved as a CSV file defined by `PROCESSED_PATH`.

Finally, in the virtual agent mode, three additional configuration options are used. `PARTICIPANT_CSV_DIR` specifies the path to the directory containing participant response tables; each table includes a column with ground truth classifications, the name of which is defined by `TARGET`. The participant being simulated in a given run is identified by the `PARTICIPANT_TO_MODEL` parameter, which defines the participant ID whose data is used to emulate classification behavior. This virtual agent setup is explained in detail in Section 2.2.

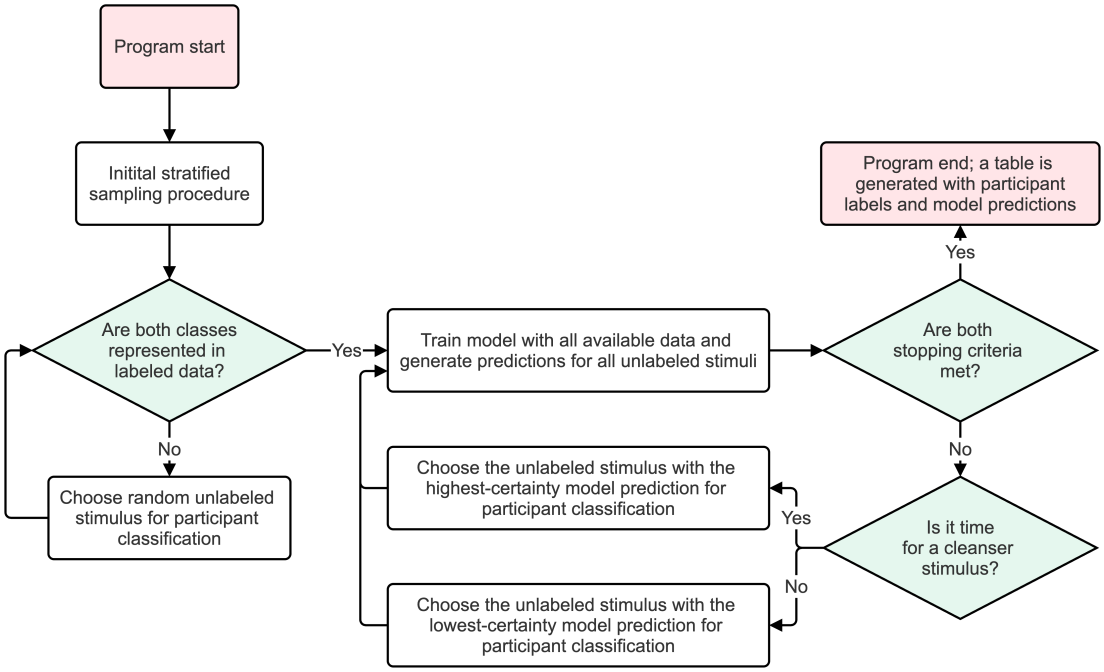

Figure 2.1: Stages of the AsTRiQue's runtime. The program starts with the initial stratified sampling procedure. Before moving on, class diversity is ensured. After an initial model is built, enough samples are collected (using uncertainty sampling with cleanser stimuli when desired), rebuilding the model after every participant answer. Once the two requirements are met (minimum iterations and model certainty cutoff criteria), the runtime ends.

2.1.3 Program Runtime

After the framework is configured and launched, it runs iteratively in several stages (depicted visually in Figure 2.1). The first stage of the runtime begins with an initial sampling procedure based on spatially stratified sampling as introduced in 2.1.2.

This approach differs significantly from simple random sampling. While random sampling may over-represent densely clustered areas of the predictor space and entirely miss sparser regions, stratified sampling guarantees that the entire predictor space is uniformly covered. This initial diversity provides a more representative overview of the distribution of stimuli and their associated perceptual properties. As a result, the algorithm begins its learning process with a more balanced and informative training set.

The participant is then asked to classify each of the selected stimuli. After collecting these initial judgments, the system checks whether the resulting labeled

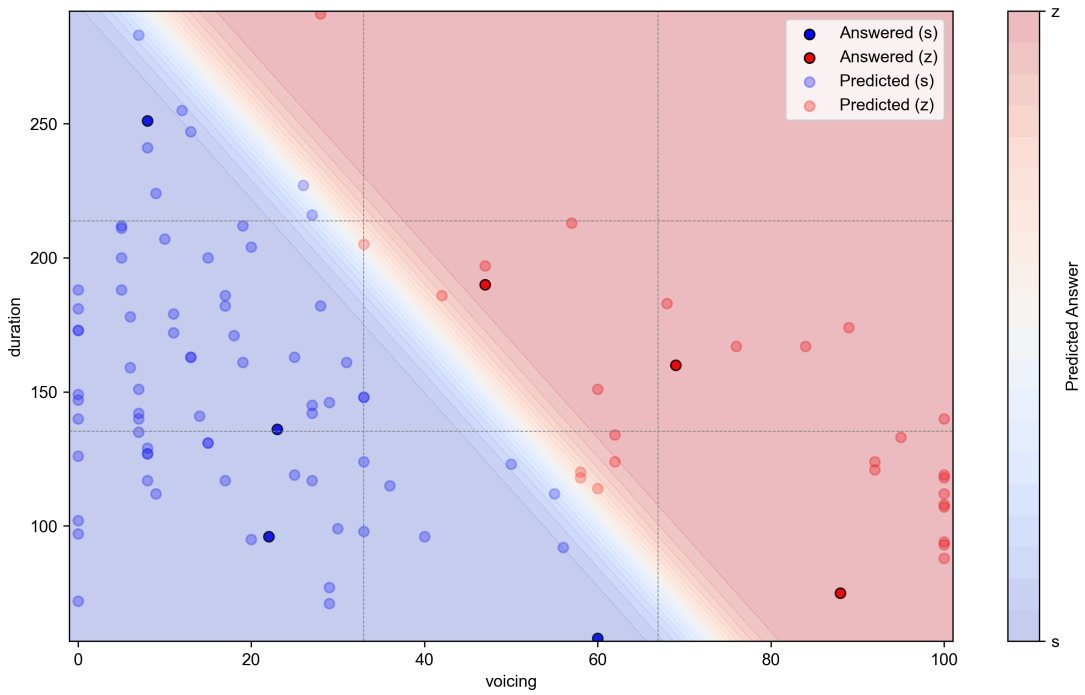

Figure 2.2: The state of the model after the initial stratified sampling procedure: the first model was built on 7 data points from different squares of a 3 by 3 grid. Two squares remain entirely unpopulated by stimuli, and thus could not be taken into account in the sampling process. (Showcase using data from Kocjančič and Bořil, 2025 as discussed in Section 2.2)

set includes examples from both target classes. This step is crucial: without class diversity, a binary classifier such as logistic regression cannot function. If all initial responses fall into a single class, the system enters a fallback mode, drawing additional samples at random until at least one sample from each class is obtained.

Once class diversity is ensured, the first logistic regression model is trained. This model provides both predicted classifications and an associated certainty measure for each unlabeled stimulus. These certainty values are derived from the predicted probabilities assigned by the model to each class. At this point, the initialization phase concludes, and the algorithm transitions into the AL loop. An example state of the model after the initial sampling stage can be seen in Figure 2.2.

After the initial model is built, the main loop begins. The core of the AL procedure is based on uncertainty sampling, a well-established strategy in which

the system selects the stimulus it is least certain about (Settles, 2012, p. 11–18). At each iteration, the current logistic regression model is used to calculate predicted probabilities for all stimuli that the participant has not yet classified. The stimulus whose prediction certainties for each of the two classes are the closest to 0.5 (i.e., the point of maximum uncertainty) is selected and presented to the participant for classification.

This sampling strategy is computationally efficient and easy to implement, making it well-suited for the type of lightweight experimental frameworks typically used in phonetic research. While more sophisticated strategies such as query-by-committee (R. Wang et al., 2012; Seung et al., 1992) or expected model change maximization (Cai et al., 2017, 2013) exist, these go beyond the scope of this thesis and could certainly be subject of interesting research in the future.

After each new classification, the model is retrained using the updated set of labeled data, and the predictions and certainty values for unlabeled data are recalculated. This loop continues until both stopping criteria are met: when the minimum number of iterations determined by the minimum iterations parameter is met and when all remaining unlabeled stimuli have predicted classifications with certainty values exceeding the model certainty cutoff threshold, the experiment terminates. At this point, the model is considered sufficiently confident to predict the participant's likely responses for the remainder of the dataset, and no further data collection is needed.

To mitigate the cognitive strain that can result from repeated exposure to borderline or ambiguous stimuli, the system incorporates an optional "cleanser" mechanism during the main loop. Every nth iteration after the initial stratified sampling and class diversity-ensuring random sampling procedures, as defined by the cleanser frequency parameter, the system presents the participant with the highest-certainty stimulus instead of the most uncertain one. These cleanser stimuli are typically easy to classify and can furthermore serve as perceptual

anchors, helping participants maintain a consistent internal frame of reference throughout the experiment.

This design choice is grounded in the informal observation that participants' perceptual judgments can become unstable when faced with a continuous stream of ambiguous stimuli. Providing occasional high-certainty items acts as a form of cognitive reset, reducing frustration and encouraging sustained engagement. While this is the only feature of this sort currently implemented in AsTRiQue, future versions of the framework could explore additional forms of distraction or variation, such as distractors unrelated to the experiment or more complex forms of stimulus ordering.

2.2 Developing and Evaluating the Framework

In order to assess AsTRiQue's practical utility and performance, it is necessary to test it in conditions comparable to how a real perceptual experiment would be run using the framework. This allows for testing of both the sampling strategy and the learning model's ability to approximate participant behavior using a limited subset of responses. However, evaluating such a system presents a number of challenges, particularly when considering the involvement of human participants during the development and debugging phase.

For the evaluation in this thesis, research data from a pre-existing experiment was used and the data for evaluation obtained using virtual agents, which act as stand-ins for human participants (and, as such, take over the role of the oracle). In the context of this framework, a virtual agent is a fully deterministic response model constructed from the pre-labeled data; essentially a lookup table (though other approached to virtual agents have been explored: e.g., Einfeldt et al., 2024). When queried with a given stimulus, the agent returns the corresponding pre-recorded answer of the participant it is simulating.

One practical challenge in evaluating the framework lies in the selection of configuration parameters. The framework includes a wide range of tunable settings:

grid resolution for initial stratified sampling, model certainty cutoff thresholds, minimum iterations, and cleanser frequency. Exploring all combinations of these parameters across a varied dataset is, in the scope of this thesis, infeasible. This thesis therefore provides a brief analysis of various configuration combinations with a deeper analysis under a subset of these configurations deemed reasonable to demonstrate its basic functionality.

A dataset derived from an existing experiment by Kocjančič and Bořil (2025) was selected to explore the framework's capabilities. This study investigated Czech children's production of sibilants and the corresponding adult perceptual judgments of voicing. Specifically, the perception experiment involved a four-alternative forced-choice task using stimuli spanning /s/, /z/, /ʃ/, and /ʒ/ along a continuum defined by the percentage of voicing and duration of each segment in milliseconds. The data set contains 104 selected stimuli with voicing ranging from 0% to 100%, representing natural productions by children aged 3 years and 5 months to 6 years and 5 months. These stimuli were judged by 31 adult listeners, and their perceptual labels were recorded.

To adapt the data for evaluation within the AsTRiQue framework (which is currently restricted to binary classification tasks) the stimuli were, after careful consideration, grouped into two categories: the unvoiced fricatives /s/ and /ʃ/, and the voiced fricatives /z/ and /ʒ/. This reduction allows the use of a two-alternative forced-choice paradigm, which aligns with the system's capabilities. Each stimulus is also associated with the two aforementioned numeric predictors derived from acoustic measurements.

Two datasets have been constructed for evaluation. The first, exploratory dataset has been created by running a virtual agent simulation for each of the 31 virtual agents simulating the participants with one runtime for each a set of configurations covering the entire spectrum of possibilities: stratified sampling resolution with all (integer) values between 1 and 10, minimum iterations between 0 and 100 in steps of 10, and model certainty cutoff between 0.05 and 1 in steps

of 0.05, including each of the bounding values listed. 31 participants, 10 stratified sampling resolution values, 11 minimum iteration values, and 20 model certainty cutoff values yield 68200 data points in the exploratory dataset.

Clearly, certain configurations would not be very useful in a real-world setting (such as model certainty cutoff values of 1, requiring every sample to be labeled by the participant, as the model should never produce a certainty value of 1.0; or high minimum iteration values, creating a hard limit on the usefulness of applying the AL framework). For this reason, a second, in-depth dataset was created, in which only a pre-selected finetuned subset of possible configurations was included. The following configurations were included in this dataset: stratified sampling resolution with all (integer) values between 2 and 10, minimum iterations between 0 and 90 in steps of 5, and model certainty cutoff always set to 0.95, again including each of the bounding values listed. 31 participants, 9 stratified sampling resolution values, and 19 minimum iteration values yield 5301 combinations, each of which was simulated 10 times, yielding 53010 data points in the in-depth dataset. Both datasets are available for download at the address in Appendix B.

An analysis of both datasets has been conducted, the results of which will be presented in Section 3. The exploratory dataset offers a basic overview of how the framework behaves in various situations, while an analysis of the in-depth dataset yields more reliable, close-up information about how it fares in situations more representative of real-world applications.

This data is moderately well-suited for testing the system in a realistic scenario. However, it is crucial to emphasize that performance on this experiment is not necessarily indicative of how the system would perform on other perceptual tasks. In this case, the distribution of stimuli and their associated labels follows a largely linear pattern in the acoustic-perceptual space; a structure that aligns well with the assumptions of logistic regression, upon which the framework is currently based. Experiments with more complex distributions of data may produce poorer performance unless the model architecture is adjusted accordingly.

Conversely, when an experiment is designed with AL in mind from the beginning, the framework could potentially yield much better results. These possibilities, along with broader implications for experimental design, are explored further in Section 4.3.

2.2.1 Technical Implementation

This section focuses on selected functionalities of the framework. The full codebase is open-source under the GPL-3.0 license (Free Software Foundation, 2007) and can be accessed via the GitHub repository referenced in Appendix A. Readers are encouraged to consult the source code alongside this section to gain a more comprehensive understanding of the implementation details.

The framework's core logic is implemented as a set of modular Python functions operating on `pandas.DataFrame` objects containing stimulus metadata and classification results. At runtime, the system first initializes a dataframe with the correct columns.

Stimulus selection begins with `get_stratified_samples(...)`. Stratified sampling is implemented by dividing the predictor space into an $n \times n$ grid using `pandas.cut()` , assigning each stimulus to a bin according to its `predictor1` and `predictor2` values, and sampling one item at random from each non-empty bin.

Model training is handled by `train_model(...)`. Labeled samples (rows where a `participant_classification` value is present) from the dataframe with the metadata are used for fitting a logistic regression model from scikit-learn. Predictions and their respective certainty values for unlabeled items are obtained with `model.predict()` and `model.predict_proba()`, storing the most likely class and the associated prediction certainty in the dataframe. The certainty value is defined as the model's predicted probability for the assigned class (after training, `predict_proba()` returns the probability of each of the two classes for all unlabeled items; the higher of the two values is selected for each stimulus).

This value, representing the model's confidence in its prediction, is stored in the dataframe for later use in sampling and stopping criteria.

During the active learning loop, `get_sample(...)` performs sample selection. If the optional cleanser mechanism is active and the current iteration index matches the configured interval, the function returns the highest-certainty unlabeled stimulus. Otherwise, it returns the lowest-certainty item. This stimulus is then passed to the oracle (either a live participant or a virtual agent) for labeling via the `query_participant_classification(...)` function, after which the main dataframe is updated.

When evaluation is required after the program runtime ends, the oracle is queried for labels of all remaining unlabeled stimuli using `evaluate_model(...)`, which are then stored in the dataframe, and compared to model predictions using `accuracy_score()`, `confusion_matrix()`, and `classification_report()`.

Visualization is performed by `plot_results(...)`. This function uses Matplotlib to plot labeled and predicted points in the predictor space, overlay a model certainty contour derived from the model's `predict_proba()` output, and draw the stratified sampling grid. Colors and transparency distinguish between oracle labels and predictions.

Finally, `export_data(...)` writes the processed dataframe to a CSV file, after clearing prediction columns for already labeled rows to avoid conflating actual and inferred data. This modular structure allows the same implementation to support both live and simulated workflows with minimal code changes (the implementation of the `query_participant_classification(...)` function being the main difference), while keeping the learning, sampling, evaluation, and visualization stages unified.

2.2.2 Live Showcase

While the system will not be ready for real-world applications with real participants until further development and evaluations are completed, a basic online showcase

has been constructed to display its capabilities. Separate Jupyter notebooks are available to examine the live participant workflow (essentially a usable perception experiment using AsTRiQue) and virtual agent mode, both including an evaluation feature to see concrete statistics about performance. Both can be accessed from the project's GitHub repository linked in Appendix A.

3 Results

3.1 Data Analysis

The results presented in this section are based exclusively on the evaluation using the dataset from Kocjančič and Bořil (2025). While this dataset provides a controlled test case for the system, it should be noted that its structure is not necessarily representative of other perception experiments. For one, the stimulus–response space in this dataset exhibits a largely linear separation between categories, which aligns with the assumptions of the logistic regression model employed. Performance on experiments with more complex distributions may differ substantially. For this reason, the number of iterations saved (items for which the system made predictions without requiring participant input), is expressed as an absolute number out of the 104 stimuli in the dataset, rather than as a percentage, to avoid misleading generalizations.

3.1.1 Exploratory Dataset

The exploratory dataset was used to obtain an overview of the influence of different configuration parameters on the framework's performance. Correlational analyses (Pearson correlation, significance threshold set at $\alpha = 0.05$) confirm the expectation that prediction accuracy is positively associated with all three configuration parameters: stratified sampling resolution ($r = 0.1077$, $p < 0.001$), minimum iterations ($r = 0.6140$, $p < 0.001$), and model certainty cutoff ($r = 0.1829$, $p < 0.001$). At the same time, these parameters each had a negative effect on the number of iterations saved: stratified sampling resolution ($r = -0.1609$, $p < 0.001$), minimum iterations ($r = -0.7726$, $p < 0.001$), and model certainty cutoff ($r = -0.3163$, $p < 0.001$). In other words, increasing these settings tends

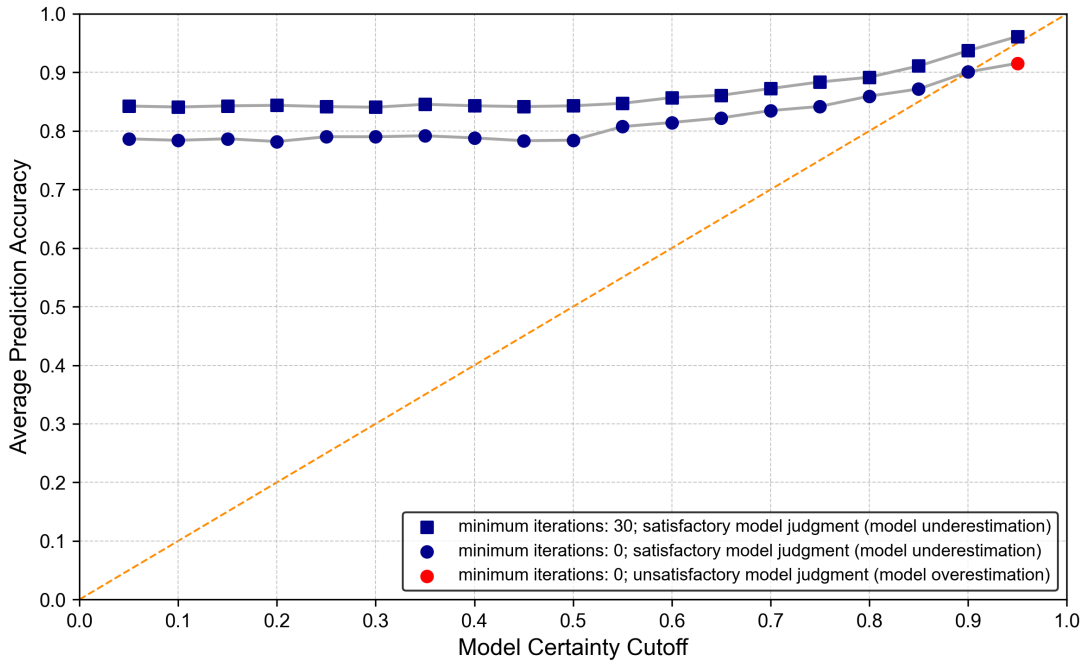

Figure 3.1: Average model prediction accuracy based on model certainty cutoff for minimum iterations equal to 0 and 30 using the in-depth dataset. An orange identity line ($y = x$) indicates perfect calibration.

to improve prediction accuracy at the cost of requiring more participant-labeled trials.

The evaluation therefore constitutes an optimization problem with two competing objectives: maximizing accuracy while minimizing the number of required participant responses. This trade-off is well captured by the concept of a Pareto front; this will be further discussed and illustrated using the in-depth dataset in Section 3.1.2.

Figure 3.1 compares average prediction accuracy with the model certainty cutoff for two values of minimum iterations (0 and 30). The orange identity line indicates the point at which the certainty measure is, on average, perfectly calibrated to actual accuracy. If the certainty metric matches reality, the average prediction accuracy after the stopping criterion should always be equal to or higher than the model certainty cutoff value: since the runtime terminates only when *all* prediction certainties exceed the preset cutoff, the final accuracy is necessarily the mean of a set of values each equal to or greater than the cutoff. Points below

the line thus indicate overestimation of accuracy (overconfidence), whereas points above indicate a satisfactory result (and possibly even underestimation).

For minimum iterations set to 0, the final point at a model certainty cutoff of 0.95 falls below the identity line, suggesting an unsatisfactory result on average. This is likely due to premature stopping; the system terminates after a small number of iterations, having built a model with high (but inaccurate) prediction certainty values, without having gathered sufficient evidence. By contrast, with minimum iterations set to 30, the final point lies above the line, indicating that the model is well-calibrated or slightly underconfident. In practice, such post hoc observations are not available during the design of a new experiment (with the exception of pilot runs), but they illustrate that setting a non-zero minimum iterations parameter (for this data, at least 30) can safeguard against overconfidence in early stopping.

3.1.2 In-Depth Dataset

The in-depth dataset focuses on a narrower, more realistic range of configurations to obtain a closer view of the framework's performance in plausible experimental conditions. Within this subset, the trade-off between prediction accuracy and iterations saved can be examined in greater detail. In the context of such competing objectives, multi-objective optimization provides a useful interpretive framework: the Pareto front represents the set of configurations for which no improvement in one objective can be achieved without compromising the other (Miettinen, 1998, p. 11). Here, each configuration can be plotted in a two-dimensional space, with prediction accuracy on one axis and iterations saved on the other (3.2). Points lying on the Pareto front thus denote optimal compromises; shifting towards higher accuracy necessarily entails sacrificing iterations saved, and vice versa. The choice of an operating point therefore depends on the experimenter's priorities: a configuration closer to the accuracy extreme may be justified when precision is paramount, whereas one favoring iterations saved may be preferred when reducing

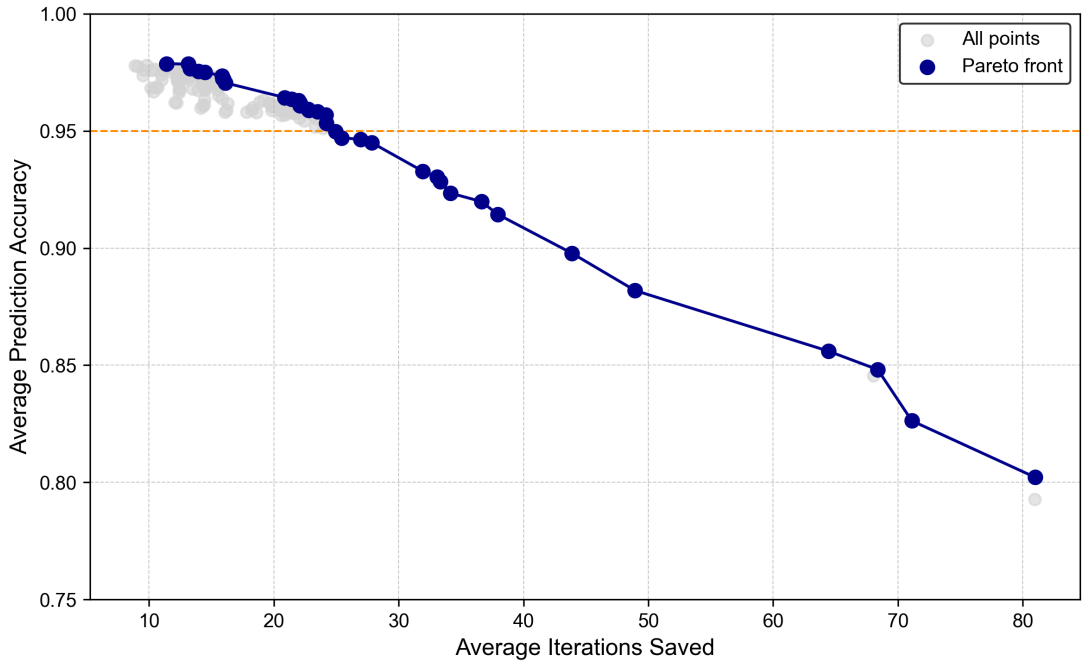

Figure 3.2: Pareto front of various minimum iterations and stratified sampling resolution configurations at a model certainty cutoff of 0.95, represented by the dashed orange line. Each point represents a Pareto optimal combination of these two values; these configurations and their respective performances are listed in Appendix C.

participant workload is the primary aim. It should be stressed, however, that the Pareto front observed here reflects only the specific data analyzed in this thesis and should not be interpreted as a general guideline for other experimental contexts.

As shown in Figure 3.3 and first discussed in Section 3.1.1, low values of minimum iterations are associated with a slight but consistent tendency towards overconfidence: average accuracies fall marginally below the model certainty cutoff of 0.95. This again reflects premature stopping after too few labelled trials. From a practical perspective, a value of at least 20 appears to avoid this effect in the present dataset, yielding accuracies at or above the cutoff.

Figure 3.4 examines the effect of stratified sampling resolution when minimum iterations are fixed at 20. Very low resolutions lead to a modest decline in calibration, with points falling just below the cutoff line. This suggests that too coarse a sampling grid in the initial selection can limit the representativeness of the early training data, again allowing for overconfident model predictions. In

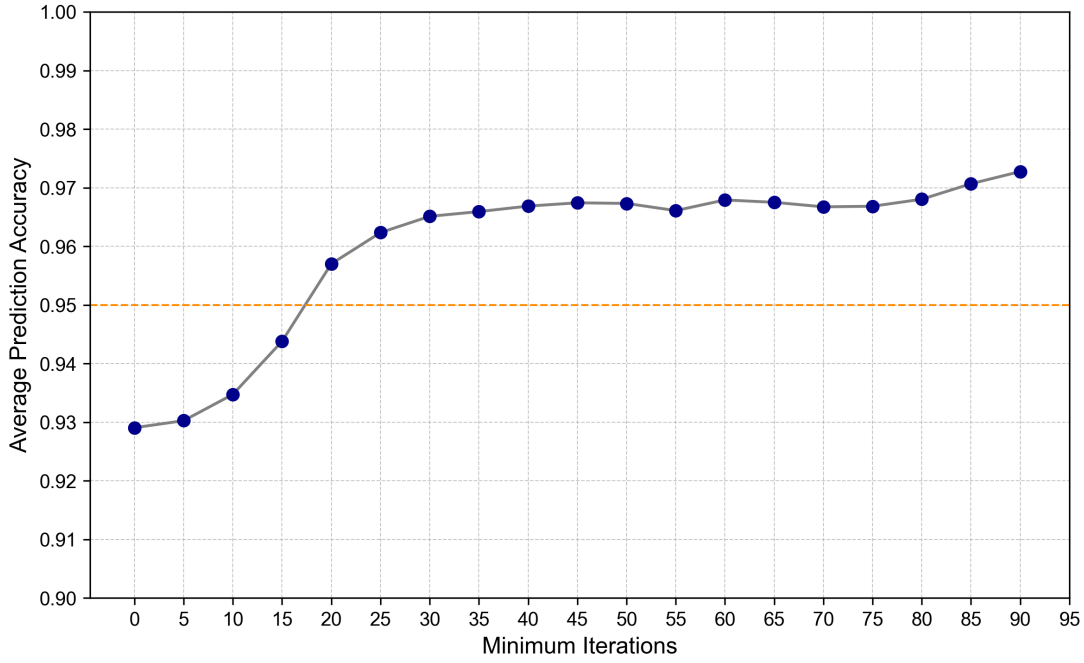

Figure 3.3: Average model prediction accuracy based on the configured minimum iterations using the in-depth dataset. A dashed orange line at $y = 0.95$ displays the model certainty cutoff value.

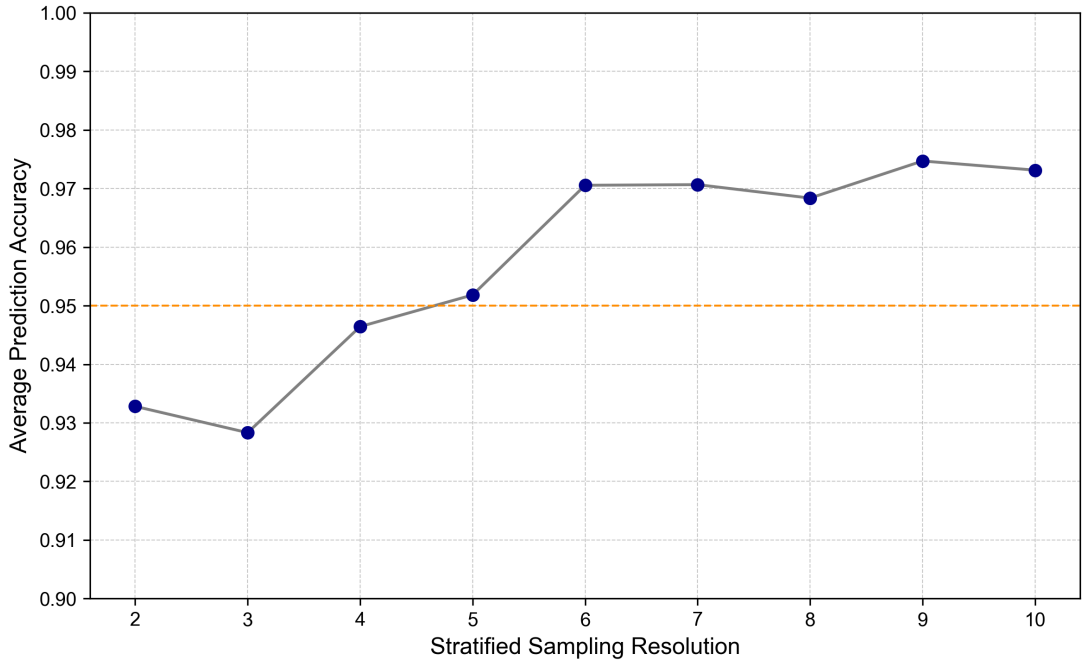

Figure 3.4: Average model prediction accuracy based on stratified sampling resolution for minimum iterations equal to 20 using the in-depth dataset. A dashed orange line at $y = 0.95$ displays the model certainty cutoff value.

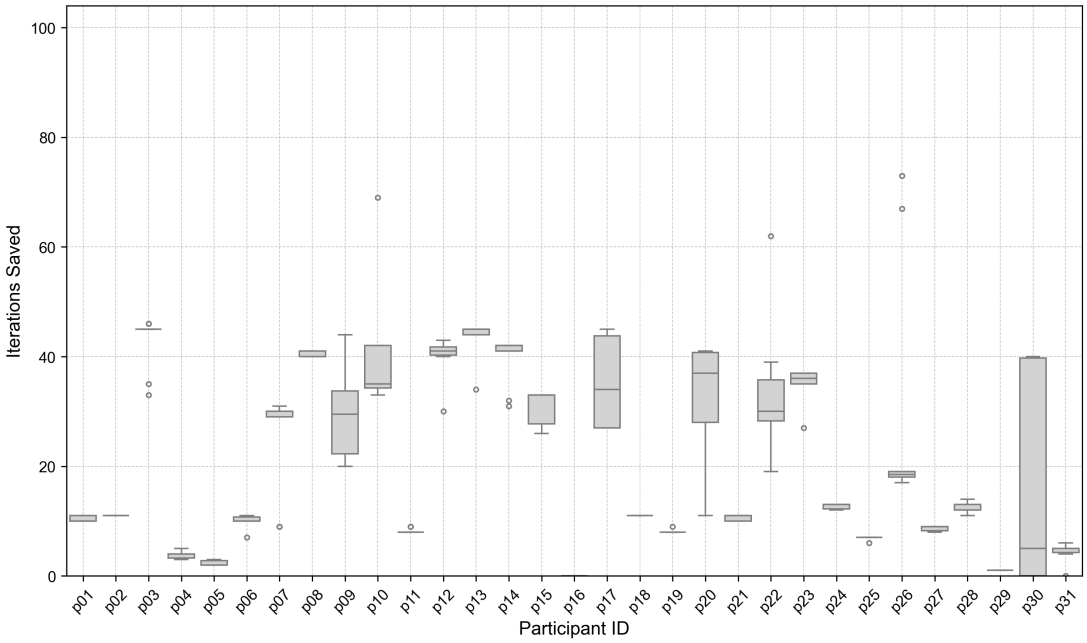

Figure 3.5: This chart displays the amount of iterations saved for each simulated participant in the 10 runtimes of the framework with minimum iterations set to 20 and a stratified sampling resolution of 5.

the present dataset, a resolution of at least 5 (19 stimuli in total; 6 of the 25 spaces created by the 5 by 5 sampling grid were unoccupied by data) provides stable results across the range of tested minimum iterations. As with minimum iterations, this reflects the need to balance parameter settings: ill-considered configurations can erode the accuracy gains that active learning seeks to preserve.

It must also be mentioned that the performance varied among the simulated participants. Figure 3.5 illustrates this phenomenon: while the system struggled to reduce the experiment's length by just a few stimuli for some virtual agents, for others, it was able to consistently save over 40. This can be explained by oracle noise, which is discussed in Section 4.4.

3.2 Using AsTRiQue with Custom Data

The AsTRiQue framework is fully modular and can be applied to any suitably formatted dataset. Researchers may replace the supplied stimuli and metadata with their own, provided that the dataset includes two numeric predictors and binary target labels. Detailed instructions for preparing and loading custom

data are provided in Section 2.1.2, and the complete source code, including example datasets and configuration templates, is available in the project's GitHub repository (Appendix A).

4 Discussion

This section examines the broader implications of the system described in this thesis, its limitations, and potential directions for future research. While the results demonstrate promising reductions in participant workload with minimal accuracy loss, several caveats must be addressed. These concern both the theoretical groundwork of the AL framework and the practical realities of its use in phonetic perception experiments.

4.1 Suitability for Task Types

The system currently supports only binary classification within 2AFC paradigms; this decision was made for reasons of simplicity of implementation. However, this excludes a range of other experimental designs commonly used in phonetics. These more complex paradigms could, in theory, benefit from AL as well, but extending the system to support them would require both conceptual and technical changes. For example, supporting continuous ratings would necessitate a regression-based AL model and a reevaluation of how informativeness is quantified. While this is outside the current scope, it presents a rich opportunity for future development.

4.2 Assumptions about Data Distribution

As implemented, the system assumes that the relationship between stimulus properties and perceptual categories is approximately linear. This assumption arises from the use of logistic regression as the learner model. In experiments where the data distribution deviates significantly from linear separability, the model may underperform. This is not a fundamental limitation of AL per se, it is rather a function of model choice. The framework is designed to allow different learner

algorithms to be swapped in with minimal code adjustments; as such, more flexible classifiers such as support vector machines or neural networks could be used in future experiments. However, more complex models may require a greater number of training examples before achieving acceptable predictive performance, thereby reducing the efficiency gains that make the proposed AL approach desirable in the first place. For experiments with clearly non-linear category boundaries, this trade-off will need to be weighed carefully.

4.3 Considerations about Evaluating the System

Evaluation in this thesis relies on virtual agents, deterministic models based on past participant responses. While this approach allows rapid and repeatable testing under controlled conditions, enabling the identification and resolution of technical issues without the delays inherent in research with human participants, it cannot fully replicate human behavior. Virtual agents do not experience fatigue, attention shifts, or perceptual recalibration over time. Consequently, certain aspects of the framework, most notably the efficacy of the cleanser mechanism, cannot be meaningfully assessed in the current setup. The inclusion of high-certainty stimuli at regular intervals is intended to provide perceptual anchors and mitigate the destabilizing effect of repeated exposure to ambiguous items. This cognitive reset may help participants maintain consistent judgment criteria throughout the experiment; however, without empirical testing with human participants, this remains hypothetical.

Furthermore, virtual agents cannot model order effects. A large part of perception experiments is run with stimuli in random or pseudo-random order to minimize these (e.g., Svatošová and Bořil, 2023; Z. Wang and Gobl, 2023; Sheppard et al., 2017; Franco et al., 2011). However, AL-based systems inherently disrupt this: multiple similar, ambiguous stimuli may (in fact, if the program is working correctly, *should*) be presented one after another, potentially inducing local perceptual hysteresis (unless mitigated by the aforementioned cleanser stimuli

or other strategies). Such effects may lower data quality or introduce noise; these interactions merit further empirical study.

The present evaluation is also limited in scope, as it is only based on a single experiment in which the stimulus space had a largely linear perceptual structure, aligning well with the assumptions of the logistic regression model used (this was a conscious, post hoc decision based on results of the experiment). It remains unclear how the system would perform in more complex experimental settings. Experiments with more than two categories, non-linear perceptual boundaries, or a higher number of relevant features (should the framework be adapted to these paradigms) could challenge the model's ability to achieve high fidelity with minimal queries. Conversely, when an experiment is explicitly designed with AL in mind, the system may outperform traditional paradigms. Performance would be dependent on a multitude of factors that are beyond the scope of AsTRiQue in its current state.

A thorough evaluation of performance with the aforementioned changes implemented would ideally involve a dedicated perception experiment specifically constructed for this purpose. In such an ideal scenario, a new 2AFC experiment would be designed (or a suitable existing one adapted) to create a balanced and informative stimulus space. The experiment would then be run in parallel using two methods: a control condition employing the traditional fixed-trial design, and a test condition in which the stimuli are selected adaptively by the framework. Comparisons between the two conditions would allow direct measurement of the system's efficiency, predictive accuracy, and any potential influence of stimulus selection on participant responses. Based on these findings, the framework could be iteratively refined and re-tested on fresh participant groups until its behaviour in real-world use was well understood.

In practice, such an evaluation is difficult to conduct. Human participation in perception experiments is resource-intensive: each test run demands recruitment, scheduling, compensation, and potentially ethical approval. The logistical overhead

is magnified if each refinement to the system must be followed by a new round of human testing. These constraints make large-scale, iterative evaluation with live participants impractical during the development phase, reinforcing the pragmatic choice of virtual-agent-based evaluation for the present study.

4.4 Noisy Oracles and the Concept of Fidelity

Most AL work, including the proposed framework, works under the strong assumption that the quality of labeled data is high (Settles, 2009, p. 36); that is to say, each participant's answers are consistent. However, that is not necessarily always the case with human subjects. If a participant is inconsistent in their responses, the model may never stabilize, continuing to request new labels indefinitely. This undermines the efficiency gains of AL. In such cases, it may potentially even be useful to estimate participant consistency dynamically. Metrics could be devised to track internal coherence, perhaps based on agreement with the model over time or self-reported confidence. If a participant appears to be responding at random, the system might flag their responses as erratic, and the researcher running the experiment could later determine if and how they want to consider these results in their analysis.

If we are interested in finding a ground truth, rather than modeling the behavior of individual participants, AL could also be implemented in a slightly different way, where the participants (as oracles) would, as it were, work together in a multi-oracle crowdsourcing effort. Such approaches, which take noisy oracles into account, have been explored in AL research (Ni and Ling, 2012; Sheng et al., 2008), but significant effort would be required to adapt them to a paradigm suitable for perception experiments.

An important question remains, however: what should the framework be optimizing for at all in terms of prediction accuracy? For example, a metric of "fidelity" could be used, defined as the proportion of participant labels plus model predictions matching the participant's full label set (or would-be label set

in real-world applications without prior data). While intuitively appealing, this metric raises important questions. High fidelity implies that the model replicates participant behavior closely, but if participant responses include noise or error, is such replication desirable? In fact, some divergence from participant labels may indicate better generalization. For example, a participant might mislabel a stimulus due to inattention, but the model, trained on the bulk of consistent responses, assigns it "correctly" based on patterns in the participant's behavior. In such a case, the model's "error" may be preferable to the participant's own judgment.

Fidelity also obscures outlier effects. A stimulus that is genuinely atypical (i.e., an outlier) might be consistently misclassified by the model because it falls far from the main decision boundary. If such outliers are not queried during AL due to high model confidence, their labels may be imputed incorrectly. This has the effect of "leveling out" true variation in the data. Whether this is a flaw or a feature depends on the goals of the experiment. If the aim is to capture a participant's global categorization tendencies, this smoothing may be acceptable. If the aim is to preserve every individual decision, it may not be.

Finally, one must consider the practical implications of AL efficiency. The 20% reduction in required labels observed in this experiment may not seem revolutionary at first glance. However, in real-world research contexts, experimenters are often forced to make painful compromises: reducing stimulus sets, simplifying hypotheses, or collapsing categories simply to stay within the bounds of a manageable experiment duration. AL provides a principled way out of this trap. Rather than discarding stimuli before the experiment begins, researchers can include a large and richly varied stimulus set, confident that AL will reduce it to the most informative core. In this sense, the method shifts the burden of simplification from the researcher to the algorithm. This not only reduces the risk of premature assumptions but also allows the complexity of the perceptual space to be respected.

Crucially, AL can help determine how much data is actually needed. If the model stabilizes quickly, it suggests that the underlying perceptual space is simple. If not, it signals that more data is required. This feedback is of great value, especially in exploratory research. The system thus becomes not just a tool for reducing workload, but a diagnostic instrument in its own right.

5 Conclusion

This thesis has introduced and evaluated AsTRiQue, a Python-based framework for applying active learning to two-alternative forced-choice perception experiments in phonetics. The system enables adaptive stimulus selection and participant-specific response modeling, with the aim of reducing participant workload while maintaining high-quality results. By combining an initial stratified sampling phase with uncertainty-based query strategies while retaining configurable stopping criteria, the framework provides an operational demonstration of how AL could be used in the methodology of phonetic research.

The results, derived from virtual-agent simulations using existing experimental data, indicate that the framework can achieve notable efficiency gains, reducing the number of required participant-labeled trials without substantial loss in predictive performance. While the present implementation is constrained to binary classification within a logistic regression architecture, the underlying design is modular, allowing for integration of alternative learner models and sampling strategies. The evaluation further shows that the system's configuration shapes the balance between accuracy and workload reduction; this highlights the necessity of careful configuration when deploying the system.

The potential benefits of this approach extend beyond incremental efficiency. By offloading stimulus selection from the researcher to an algorithmic process informed by the model in real time, phoneticians can design experiments with richer and more varied stimulus sets, confident that the system will focus on the most informative items. By reducing the preemptive simplification of experimental materials needed, this shifts the traditional trade-off between breadth of data and participant endurance.

Nonetheless, several limitations must be addressed before real-world deployment. While the system can be deployed with human participants in a rudimentary experimental setting, the current evaluation relies entirely on virtual agents, which cannot replicate the variability, fatigue, or context effects of human perception. The handling of noisy or inconsistent responses remains a conceptual challenge, and the interaction between AL-driven stimulus ordering and perceptual hysteresis effects is yet to be studied. Furthermore, experiments with more complex category structures or non-linear decision boundaries may require more flexible learning algorithms, at potential cost to the efficiency gains observed here.

Future work should therefore focus on empirical validation with human participants, systematic exploration of alternative learner models, and extension of the framework to support additional experimental paradigms, such as multi-class classification and regression-based rating tasks. Strategies for handling noisy oracles and integrating real-time participant feedback also merit investigation. Such developments would broaden the framework's applicability and improve its robustness across more diverse research contexts.

While the system proposed in this thesis is part of the first steps in the research of AL use in phonetics (following Einfeldt et al., 2021 and Einfeldt et al., 2024), the findings suggest that it may have a promising future. Its demonstrated ability to reduce participant workload without compromising predictive accuracy is encouraging, but its broader adoption will depend on addressing the theoretical and practical concerns outlined above. With continued refinement and empirical testing, AL-based approaches such as AsTRiQue may not only optimize data collection but also reshape the way perceptual experiments are conceived.

References

Cai, W., Zhang, M., & Zhang, Y. (2017). Batch Mode Active Learning for Regression With Expected Model Change. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(7), pp. 1668–1681. https://doi.org/10.1109/tnnls.2016.2542184

Cai, W., Zhang, Y., & Zhou, J. (2013). Maximizing Expected Model Change for Active Learning in Regression. *2013 IEEE 13th International Conference on Data Mining*, pp. 51–60. https://doi.org/10.1109/icdm.2013.104

Cooper, S., & Cooper, S. (2023). Exposure-independent comprehension of Greek-accented speech: Evidence from New Zealand listeners. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 6–10.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, *20*(3), pp. 273–297. https://doi.org/10.1007/bf00994018

Einfeldt, M., Sevastjanova, R., Zahner-Ritter, K., Kazak, E., & Braun, B. (2021). Reliable Estimates of Interpretable Cue Effects with Active Learning in Psycholinguistic Research. *Interspeech 2021*, pp. 1743–1747. https://doi.org/10.21437/interspeech.2021-1524

Einfeldt, M., Sevastjanova, R., Zahner-Ritter, K., Kazak, E., & Braun, B. (2024). The use of Active Learning systems for stimulus selection and response modelling in perception experiments. *Computer Speech & Language*, *83*, p. 101537. https://doi.org/10.1016/j.csl.2023.101537

Erickson, D., Rilliard, A., Li, Y., Menezes, C., Kawahara, S., Sadanobu, T., Hayashi, R., Shochi, T., de Moraes, J., & Obert, K. (2023). Cross Cultural perception of Valence and Arousal. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 1776–1780.

Franco, A., Cleeremans, A., & Destrebecqz, A. (2011). Statistical Learning of Two Artificial Languages Presented Successively: How Conscious? *Frontiers in Psychology*, *volume 2 - 2011*, p. 229. https://doi.org/10.3389/fpsyg.2011.00229

Free Software Foundation. (2007). GNU General Public License, version 3.0. Retrieved August 13, 2025, from https://www.gnu.org/licenses/gpl-3.0.html

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, pp. 2672–2680.

Guo, H., & Wang, W. (2015). An active learning-based SVM multi-class classification model. *Pattern Recognition*, *48*(5), pp. 1577–1597. https://doi.org/10.1016/j.patcog.2014.12.009

Hanžlová, A., & Bořil, T. (2023). A Perceptual and acoustic study of melody in whispered Czech words. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 669–674.

Harris, C. R., Millman, K. J., Walt, S. J. v. d., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. v., Brett, M., Haldane, A., Río, J. F. d., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), pp. 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-84858-7

Haykin, S. S. (2009). *Neural networks and learning machines* (3rd ed). Prentice Hall.

Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley. https://doi.org/10.1002/9781118548387

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, *9*(3), pp. 90–95. https://doi.org/10.1109/MCSE.2007.55

Jo, T. (2021). *Machine Learning Foundations: Supervised, Unsupervised, and Advanced Learning*. Springer International Publishing AG. https://doi.org/10.1007/978-3-030-65900-4

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., & Willing, C. (2016). Jupyter Notebooks – a publishing format for reproducible computational workflows. In F. Loizides & B. Schmidt (Eds.), *Positioning and Power in Academic Publishing: Players, Agents and Agendas* (pp. 87–90). IOS Press. https://doi.org/10.3233/978-1-61499-649-1-87

Kocjančič, T., & Bořil, T. (2025). Voicing in Czech children's sibilants: Children's productions and adult's perception.

Li, Q., & Mok, P. (2023). A Perception Study on Voice Quality and Stance in Mandarin Chinese. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 1786–1790.

Liu, S., & Li, X. (2023). Understanding Uncertainty Sampling. https://doi.org/10.48550/ARXIV.2307.02719

Liu, X., Zhang, F., Hou, Z., Mian, L., Wang, Z., Zhang, J., & Tang, J. (2021). Self-supervised Learning: Generative or Contrastive. *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1. https://doi.org/10.1109/tkde.2021.3090866

Lung, H., Liao, S., & Chen, F. (2023). Linguistic release from masking in simulated electric hearing. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 52–56.

Mackworth, N. H. (1948). The Breakdown of Vigilance during Prolonged Visual Search. *Quarterly Journal of Experimental Psychology*, *1*(1), pp. 6–21. https://doi.org/10.1080/17470214808416738

Manker, J. (2020). The perceptual filtering of predictable coarticulation in exemplar memory. *Laboratory Phonology: Journal of the Association for Laboratory Phonology*, *11*(1). https://doi.org/10.5334/labphon.240

McGuire, G. (2021). Perceptual Phonetic Experimentation. In M. J. Ball (Ed.), *Manual of clinical phonetics* (pp. 495–506). Routledge, Taylor & Francis Group.

McKinney, W. (2010). Data Structures for Statistical Computing in Python. In S. v. d. Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Miettinen, K. (1998). *Nonlinear Multiobjective Optimization* (F. S. Hillier, Ed.; Vol. 12). Springer US. https://doi.org/10.1007/978-1-4615-5563-6

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

Murphy, K. P. (2012). *Machine learning: A probabilistic perspective*. MIT Press.

Ni, E. A., & Ling, C. X. (2012). Active Learning with c-Certainty. In D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, P.-N. Tan, S. Chawla, C. K. Ho, & J. Bailey (Eds.), *Advances in Knowledge Discovery and Data Mining* (pp. 231–242, Vol. 7301). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-30217-6_20

Olsson, F., & Tomanek, K. (2009). An intrinsic stopping criterion for committee-based active learning. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL '09*, p. 138. https://doi.org/10.3115/1596374.1596398

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, pp. 2825–2830.

Pisoni, D. B. (1973). Auditory and phonetic memory codes in the discrimination of consonants and vowels. *Perception & Psychophysics*, *13*(2), pp. 253–260. https://doi.org/10.3758/bf03214136

Reteig, L. C., Van Den Brink, R. L., Prinssen, S., Cohen, M. X., & Slagter, H. A. (2019). Sustaining attention for a prolonged period of time increases temporal variability in cortical responses. *Cortex*, *117*, pp. 16–32. https://doi.org/10.1016/j.cortex.2019.02.016

Settles, B. (2009). *Active learning literature survey* (tech. rep.). University of Wisconsin Madison.

Settles, B. (2012). *Active learning*. Morgan & Claypool. https://doi.org/10.1007/978-3-031-01560-1

Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287–294. https://doi.org/10.1145/130385.130417

Sheng, V. S., Provost, F., & Ipeirotis, P. G. (2008). Get another label? improving data quality and data mining using multiple, noisy labelers. *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 614–622. https://doi.org/10.1145/1401890.1401965

Sheppard, S. M., Love, T., Midgley, K. J., Holcomb, P. J., & Shapiro, L. P. (2017). Electrophysiology of prosodic and lexical-semantic processing during sentence comprehension in aphasia. *Neuropsychologia*, *107*, pp. 9–24. https://doi.org/10.1016/j.neuropsychologia.2017.10.023

Svatošová, M., & Bořil, T. (2023). Duration as a cue for phonological voicing contrast in whispered Czech. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 1741–1745.

The Matplotlib Development Team. (2025, May). Matplotlib: Visualization with Python. https://doi.org/10.5281/zenodo.15375714

The Pandas Development Team. (2020, February). Pandas-dev/pandas: Pandas. https://doi.org/10.5281/zenodo.3509134

Vallesi, A., Tronelli, V., Lomi, F., & Pezzetta, R. (2021). Age differences in sustained attention tasks: A meta-analysis. *Psychonomic Bulletin & Review*, *28*(6), pp. 1755–1775. https://doi.org/10.3758/s13423-021-01908-x

van Rossum, G. (1995, May). *Python tutorial* (tech. rep. No. CS-R9526). Centrum voor Wiskunde en Informatica (CWI).

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, pp. 261–272. https://doi.org/10.1038/s41592-019-0686-2

Vlachos, A. (2008). A stopping criterion for active learning. *Computer Speech & Language*, *22*(3), pp. 295–312. https://doi.org/10.1016/j.csl.2007.12.001

Wang, R., Kwong, S., & Chen, D. (2012). Inconsistency-based active learning for support vector machines. *Pattern Recognition*, *45*(10), pp. 3751–3767. https://doi.org/10.1016/j.patcog.2012.03.022

Wang, Z., & Gobl, C. (2023). Perceptual effects of aliasing distortion in glottal flow modelling. *Proceedings of the 20th International Congress of Phonetic Sciences*, pp. 1801–1805.

List of Figures

List of Tables

A GitHub Repository

AsTRiQue's entire codebase including a live showcase of the system (virtual agent and live participant modes) is accessible under the GPL-3.0 license at https://github.com/prokophanzl/AsTRiQue.

B Datasets for Analysis

Both the exploratory and in-depth evaluation datasets mentioned in Section 2.2 are available for download from the GitHub repository at https://github.com/prokophanzl/AsTRiQue/tree/main/extras/evaluation.

C Table of Performance Metrics

Minimum Iterations	Stratified Sampling Resolution	Prediction Accuracy	Average Iterations Saved
85	6	0.9787	11.4161
65	9	0.9786	13.1290
10	9	0.9766	13.2935
75	6	0.9755	13.9548
65	7	0.9751	14.4935
40	6	0.9735	15.8710
55	6	0.9722	15.8774
5	6	0.9722	15.9613
30	6	0.9705	16.1258
60	2	0.9643	20.8581
60	3	0.9636	21.4097
50	2	0.9631	21.9677
50	3	0.9620	22.0839
45	3	0.9609	22.1065
40	3	0.9593	22.7645
40	2	0.9588	22.7806
35	3	0.9583	23.5000
30	2	0.9568	24.1581
25	2	0.9533	24.2194
25	4	0.9497	24.9548
15	5	0.9471	25.4387
20	4	0.9464	26.9419
25	3	0.9451	27.8258
20	2	0.9328	31.9355
5	4	0.9304	33.0742
20	3	0.9283	33.3452
0	4	0.9235	34.1548
10	4	0.9199	36.6226
15	4	0.9144	37.9355
15	3	0.8978	43.8903
15	2	0.8819	48.9387
10	3	0.8560	64.4226
0	3	0.8481	68.3871
10	2	0.8263	71.1129
5	2	0.8022	80.9903

Table 1: Performance metrics for different configurations of stratified sampling resolution and minimum iterations at a model certainty cutoff of 0.95, represented by the dashed line (configurations below this line delivered unsatisfactory results).