



Article

Achievable Minimally-Contrastive Counterfactual Explanations

Hosein Barzekar and Susan McRoy *

Department of Computer Science, University of Wisconsin-Milwaukee, Milwaukee, WI 53211, USA;
barzekar@uwm.edu

* Correspondence: mcroy@uwm.edu; Tel.: +1-414-229-6695

Abstract: Decision support systems based on machine learning models should be able to help users identify opportunities and threats. Popular model-agnostic explanation models can identify factors that support various predictions, answering questions such as “What factors affect sales?” or “Why did sales decline?”, but do not highlight what a person should or could do to get a more desirable outcome. Counterfactual explanation approaches address intervention, and some even consider feasibility, but none consider their suitability for real-time applications, such as question answering. Here, we address this gap by introducing a novel model-agnostic method that provides specific, feasible changes that would impact the outcomes of a complex Black Box AI model for a given instance and assess its real-world utility by measuring its real-time performance and ability to find achievable changes. The method uses the instance of concern to generate high-precision explanations and then applies a secondary method to find achievable minimally-contrastive counterfactual explanations (AMCC) while limiting the search to modifications that satisfy domain-specific constraints. Using a widely recognized dataset, we evaluated the classification task to ascertain the frequency and time required to identify successful counterfactuals. For a 90% accurate classifier, our algorithm identified AMCC explanations in 47% of cases (38 of 81), with an average discovery time of 80 ms. These findings verify the algorithm’s efficiency in swiftly producing AMCC explanations, suitable for real-time systems. The AMCC method enhances the transparency of Black Box AI models, aiding individuals in evaluating remedial strategies or assessing potential outcomes.

Keywords: machine learning; interpretability; feasibility; counterfactual and contrastive explanation



Citation: Barzekar, H.; McRoy, S. Achievable Minimally-Contrastive Counterfactual Explanations. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 922–936. <https://doi.org/10.3390/make5030048>

Academic Editor: Luca Longo

Received: 9 July 2023

Revised: 27 July 2023

Accepted: 1 August 2023

Published: 3 August 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Explainable AI (XAI) can serve many different aims, including increasing users’ trust in a Black Box model [1,2] or increasing users’ understanding of how the model would behave on unseen examples [3]. One function that people would like XAI to achieve, but has yet to be adequately accomplished, is to address questions about what they can do to change the predicted outcome. This type of XAI would be useful for applications that require changes in behavior such as counseling, advising, or coaching, e.g., for health, education, or personal finance, where the perceived benefits of an action and the likelihood of success contribute to both initiation and ongoing maintenance of behavior change [4–7]. We are particularly interested in applications where people establish so-called SMART goals. SMART goals are specific, measurable, achievable, relevant, and time-bound [8,9]. Such goal setting would benefit from having a high-precision explanation, which is one that includes the enabling specific value for a variable, not just its name. (See Table 1 for some examples of high-precision explanations.) We also prefer explanations that could be generated in real-time, as part of an ongoing interaction, as this type of interaction has been shown to result in greater trust in decision support systems as compared to traditional web-based graphical user interfaces [10].

Explanations specifying instances that result in a different outcome than a given one are known as counterfactuals. Counterfactuals are the type of explanation that many people find most natural and intuitive. Indeed, research across cognitive science, philosophy, and

social science has found that people expect ‘why’ questions to be answered by explanations that are contrastive, selective, and socially interactive [11]—that is, they want answers that are solution-focused. The need for explanations that support the remediation of a poor outcome has also been supported by recent user studies of XAI, which found that queries to maximize a prediction were among the top ten types of questions that people would ask of an AI model, along with more general ‘why’-type questions [12]. Wachter et al. [13] proposed that counterfactual and contrastive explanations (CCE) would be an effective approach to XAI and recent systematic reviews show growing interest in CCE among XAI researchers [14].

Table 1. Examples of high-precision explanations generated for tabular data from [3].

Adult Income		
If		Predict
No capital gain or loss, never married		≤50 K
Country is US, married, work hours > 45		>50 K
Lending a Loan		
If		Predict
FICO score ≤ 649		Bad loan
649 ≤ FICO score ≤ 699 and \$5400 ≤ loan amount ≤ \$10,000		Good loan

For intervention purposes, it is also important to consider the feasibility of achieving a counterfactual [15], which means that the change would be possible through something that a person could choose to do. Finding a set of such actions is a special case of counterfactual explanation known as algorithmic recourse, which is also related to causal machine learning [16,17]. Achievability requires that there be a domain-specific action that includes the desired change as one of its effects, either directly, such as walking to change a physical activity variable, or through a change to a state of knowledge. For example, a doctor could order a diagnostic test to rule-in or rule-out the presence of a disease. Here, we will presume that there is a knowledge model that provides this information, which distinguishes our approach from those that try to infer feasibility from a set of observations. Because options for recourse are often quite limited, providing such a model would not be difficult, but does raise some practical concerns. For example, when only a few changes are achievable, it is not clear how often recourse is possible, how hard it will be to find suitable recourse, or how often effective recourse can be found quickly enough for use in real-time interaction. Research on human-computer interaction has shown that the ideal latency for communicating recourse, as a form of keyboard-based or mobile interaction, should be no more than 200 ms for desktop or no more than 500 ms for mobile [18,19].

We propose a two-step process for finding achievable counterfactuals satisfactory for real-time interaction: first, find high-precision explanations of the current prediction and then find achievable changes that the AI model predicts would result in the desired outcome. If successful, this approach would allow a system to address both ‘why’ and ‘what can be done’ queries for the same scenario. From a technical standpoint, this process provides explanations that are local, model-agnostic, high-precision, achievable minimally-contrastive counterfactuals—concepts we shall define in detail below.

- **Model-agnostic explanations** address how a model performs under various conditions or inputs rather than attempting to explain the inner workings of a model. In practice, this means that the approach can be used for a variety of machine learning models, including those that do not generate rules or otherwise support interpretability [1].
- **Local explanations** provide justification for the behavior of a model for a particular instance, rather than all possible instances. In tabular data, as we consider here, an instance comprises a set of features, each associated with a value from a designated set of possible values, and an outcome is a discrete class, which is a feature with a known

set of possible values that has been designated as the ‘target’. Local, model-agnostic explanations can be found by perturbing the values of features of the given test case, to identify changes that affect the target prediction [1,20].

- **High-precision explanations** are local explanations indicating both the features and the value or range of values for the feature sufficient to support a prediction. Anchors [3] are high-precision local explanations that include features that justify an outcome independent of the values that the other features might have.
- **Contrastive counterfactual explanations** are local explanations that achieve a target outcome T_2 complementary to the original outcome T_1 , by changing the value of a set of input features for an instance [21]. These features might be causally related to the outcome, but one does not need to have explicit causal information to find counterfactuals; one can achieve a good approximation using the trained AI model to formulate a prediction, given a contrastive instance [22].
- **Achievable contrastive counterfactual explanations** are local explanations that achieve a target outcome T_2 complementary to the original outcome T_1 , using only features whose values would be feasible for the subject of the decision system to change. This information is specific to the domain and task. We assume that all features that would be feasible to change, and any constraints on possible changes, are provided to the system and that the system only perturbs values respecting these constraints. (For a socially-interactive system, such as a conversational agent, the system should also have access to non-technical descriptions of the actions for achieving the desired value for a feature).
- **Minimally-contrastive counterfactual explanations** are local explanations that achieve a target outcome T_2 complementary to the original outcome T_1 , using only minimal subsets of features, such that if MCC_j is a minimally-contrastive explanation, and $MCC_j \equiv \{m_1, m_2, \dots, m_j\}$ then for any $m_i \in MCC_j$, $MCC_j \setminus \{m_i\}$ would not be sufficient to achieve T_2 . We shall take it as obvious that people would favor making fewer changes, but preferences among changes involving different features depend on individual preferences.

2. Related Work

The most closely-related work is that of Poyiadzi et al. [23] who provide an algorithm for Feasible and Actionable Counterfactual Explanations (FACE). The FACE algorithm is a variant of Dijkstra’s algorithm for finding shortest paths. The FACE approach creates a graph of instances similar to a given test case. In the graph, nodes correspond to instances and edges exist between pairs of instances that differ in the value of exactly one feature if and only if it would be feasible and actionable for the anticipated user to make the corresponding change to the value of a feature, as given by a pre-specified condition function. FACE also allows one to provide a ‘weight’ function to establish preferences over different types of changes. This approach provides local model-agnostic counterfactuals, but similarly to most approaches other than anchors, this method would only return a single data point, rather than a high-precision explanation. In addition, the authors do not reveal how long it might take to create or use the graph of instances for a specific dataset, given a limited set of actionable features.

Making sure that individuals have a say in how their personal lives are affected by machine learning is the main concern of “Actionable Recourse in Linear Classification” [24]. In this paper, the authors introduce the notion of ‘recourse’, referring to a person’s ability to influence a model’s outcome by changing ‘actionable’ inputs. The authors suggest using integer programming-based techniques that are optimized for redress in linear classification issues as a means of achieving this goal, but do not consider real-time performance.

Another related work is that of Mothilal et al. [25], which provides an approach to finding a diverse set of feasible counterfactual actions by generating a set of contrasting feasible counterexamples. Their algorithm generates new contrasting examples while assessing a diversity metric over the generated examples. Feasibility itself is determined

by the approach first suggested by Wachter et al. [13], which is based on proximity, rather than a domain model. Similarly, Karimi et al. [15] suggest an optimization approach to find a minimal cost set of actions (in the form of structural interventions) that result in a counterfactual instance yielding the favorable output; that is, it associates costs with actions (edges) rather than features, acknowledging that not all modifications of a feature may be equally acceptable. They do not provide an explicit algorithm or implementation, but suggest using the Abduction-Action-Prediction steps proposed by Pearl [22]. Thus, it is not clear under what conditions this approach might be suitable for real-time interaction.

3. Materials and Methods

The primary objective of this research is to provide a real-time, high-precision, model-agnostic method for recourse. The approach is to modify and extend the functionality of the local, model-agnostic explanation algorithm proposed in “Anchors: High-Precision Model-Agnostic Explanations” and assess its suitability for use in an interactive system. The key adaptation is to use it to find minimally-contrastive counterfactuals, using only achievable modifiable factors that are also anchors.

We focus on two main research questions: (1) when the set of features is reduced to include only plausibly actionable ones, how often can the anchor’s approach find at least one explanation that would be an effective intervention using a dataset suitable for algorithmic recourse? and (2) under these circumstances, how quickly can we find such recourse? Secondly, we also assess the impacts of varying the value of an internal parameter of the anchor’s algorithm controlling how many alternatives it keeps under consideration when trying to maximize coverage (beam size).

3.1. Algorithm for Generating Achievable Counterfactuals

In the pursuit of finding counterfactuals, our approach explores alternatives to the original instances by modifying one feature at a time and then assessing the outcome using the trained model. The primary objective of this methodology is to pinpoint a variant of a given instance that prompts a shift in the classifier’s original suboptimal prediction to a more optimal one, given the context of the domain and user preferences.

Our focus on achievable minimally-contrastive counterfactuals (AMCC) led us to adopt Breadth-First Search (BFS), a graph search algorithm that explores all nodes at the current depth level before proceeding to the next. We tailored BFS to suit our objective, creating a variant termed “BFS-AMCC”, as detailed in Algorithm 1. As is common with AI search algorithms, BFS-AMCC does not start with a prebuilt graph, rather it generates nodes one at a time, for each of the allowed types of modifications. It can be used to find a single successful counterfactual, using the least number of modifications, or multiple ones, when allowed to continue searching until it either exhausts all possibilities or reaches a preset stopping condition.

BFS-AMCC initiates its operation by populating a queue with the original instance. During the algorithm’s core loop, an instance is dequeued for prediction. If the modified instance’s prediction deviates from the original, it is deemed a successful modification and the algorithm halts. If not, the algorithm proceeds to identify features for potential modification. In the absence of any specifically provided indices, the algorithm treats all categorical indices as modifiable.

BFS-AMCC also accommodates a set of ‘transition_rules’ for the features. These are optional and can be used to define valid transitions between feature states, ensuring that the algorithm only considers achievable and realistic modifications to the instances. If a ‘transition_rule’ is specified for a feature, the algorithm only considers the new feature value if the transition rule returns true when applied to the current and new feature values. These transition rules would be domain-specific, ideally derived from an ontology, but for simple data can be specified directly.

In its iterative process, BFS-AMCC sifts through each index, creating a modified instance for each valid feature value perturbation and queuing it. This iteration continues

until a modified instance prompts a shift in the original prediction to the desired outcome, or until the queue is exhausted.

Ultimately, BFS-AMCC returns the successfully modified instance—an achievable, minimally-contrastive counterfactual that influences the classifier’s original prediction towards a preferred outcome. If no successful modification is found, the algorithm returns ‘None’. This methodology guarantees the discovery of a locally-minimal counterfactual, offering advanced interpretability due to its minimal feature divergence from the original instance.

Algorithm 1 BFS for achievable minimally-contrastive counterfactuals to change the predicted value

```

1: procedure BFS-AMCC(instance, classifier, original_prediction, categorical_names, specific_indices, ignore_indices, transition_rules)
2:   queue ← [(instance, [])]           ▷ Initialize the queue with the original instance
3:   best_modified_instance ← None
4:   while queue ≠ ∅ do
5:     current_instance, changed_indices ← queue.pop()
6:     current_prediction ← classifier.predict(current_instance)
7:     if current_prediction ≠ original_prediction then
8:       best_modified_instance ← current_instance
9:       break
10:    if specific_indices ≠ None then
11:      indices_to_modify ← specific_indices
12:    else
13:      indices_to_modify ← all indices in categorical_names
14:    for index ∈ indices_to_modify do
15:      if index ∉ changed_indices then
16:        for value ∈ categorical_names[index] do
17:          if value ≠ current_instance[index] then
18:            if transition_rules ≠ None and index in transition_rules then
19:              if not transition_rules[index](current_instance[index], value) then
20:                continue
21:              modified_instance ← current_instance with index set to value
22:              queue.append((modified_instance, changed_indices + [index]))
23:  return best_modified_instance

```

3.2. Procedure for Evaluating Counterfactuals

Our empirical evaluation strategy, presented as Algorithm 2, is designed to assess the performance of our approach to counterfactual generation—specifically, BFS-AMCC. The main objectives are to calculate the proportion of successful counterfactual instances and to measure the computational time required for each successful instance. These metrics are stored in a Python dictionary for analysis.

To increase the efficiency and practicality of our approach, we introduce a 5 s time limit for each instance using Python’s ‘signal’ module. This time limit excludes extreme outliers and maintains a reasonable balance between exhaustive search and computation time. It is also very conservative, given the target for real-time interaction should be 200 to 500 ms [18,19]. In practice, this time limit is rarely ever reached, usually when there is no possible recourse.

The algorithm initializes counters for successful and failed instances, along with a timer to track the computation time for each instance.

The main loop of the evaluation algorithm processes each instance in the provided ‘suboptimal_instances’ list, which contains instances that might benefit from remediation. For each such instance, the algorithm retrieves the true label and the classifier’s prediction.

If the classifier’s prediction matches the true value, the algorithm uses the ‘AnchorTabularExplainer’ to identify high-precision features. These features are then used in the BFS-AMCC process to guide the generation of modified instances.

Next, the algorithm applies the BFS-AMCC function (Algorithm 1) to generate a modified instance within the provided time limit. If a modified instance is found within this limit, the algorithm updates the success counter, records the time taken, and logs the changes. Otherwise, if a suitable counterfactual cannot be found within the time limit, a 'TimeoutError' is raised, and the algorithm moves to the next instance, increasing the failure counter.

Upon processing all instances in the 'suboptimal_instances' list, the algorithm returns the 'metrics' dictionary. This dictionary provides a comprehensive summary of the performance of our strategy, containing the number of successful and failed instances, total computation time, and a detailed log of changes for each successful counterfactual instance.

Algorithm 2 Evaluation of Counterfactual Generation

```

1: procedure COUNTERFACTUAL_EVALUATION(ignore_indices, transition_rules, thresh_prob)
2:   Input: DatasetLoader, ML_Model, predictor, explainer, metrics, suboptimal_instances, transition_rules
3:    $num\_success, num\_failures, total\_time \leftarrow 0$ 
4:   for each  $instance \in suboptimal\_instances$  do
5:     Obtain original instance, label, and prediction
6:     if  $original\_instance\_label == original\_prediction == 1$  then
7:       Apply explainer on the original instance
8:       Extract feature indices and specific indices
9:       Set time limit to "s" seconds
10:      Start timer
11:      try:
12:         $modified\_instance \leftarrow$  BFS-AMCC
13:      except TimeoutError:
14:        Print error and continue
15:      finally:
16:        Remove time limit
17:        Stop timer and update  $total\_time$  and metrics
18:      if  $modified\_instance \neq None$  then
19:        Update metrics with success case
20:        Compute changes_for_instance and update metrics
21:      else
22:        Update metrics with failure case
23:   return metrics

```

4. Experiments and Results

This section describes each of the steps in our evaluation along with their results. The evaluation assesses the difficulty in finding an AMCC explanation, as both success rate and time taken, for a representative data set. We also perform preliminary experiments to create an optimal classifier and experiments where we vary a key parameter within the Anchors algorithm, 'beam size'. Lastly, we provide examples of the AMCC explanations that would be provided for a sample of test cases, along with their computation time.

4.1. Dataset Selection and Preparation

We selected a widely-available tabular dataset, HMEQ, that appears to include several actionable features. HMEQ is from the domain of lending, where the predicted outcomes are $\{GOOD, BAD\}$, where a bad loan is one that resulted in a default or serious delay in repayment, and a good loan is one that was repaid on time. This dataset is available from Kaggle.

This dataset has a total of 5960 instances corresponding to individual home equity loans, where the adverse outcome, $f = 1$ (BAD) occurs in about 20% of cases ($n = 1189$). For each applicant, 12 input variables were recorded; we determined that seven of them were actionable, as shown in Table 2, along with recourse actions for those variables. We noted several cases where transitions would likely only be desirable or achievable in one

direction and thus will treat MORTDUE, DELINQ, NINCQ, and DEBTINC, as feasible to decrease, but not to increase.

Table 2. Actionable features in the HMEQ dataset including recourse actions.

Feature	Description	Example Recourse Actions
LOAN	the amount of the loan request	increase (or reduce) the loan request
MORTDUE	the amount due on the existing mortgage	pay down the existing mortgage
VALUE	the current value of property	have the property reappraised
DELINQ	the number of delinquent credit lines	pay off delinquent credit
NINCQ	the number of recent credit inquiries	reduce credit applications
CLNO	the total number of credit lines	increase (or reduce) open credit lines
DEBTINC	the client's current debt-to-income ratio	pay off some debt

Note that most of the features in HMEQ have continuous numeric values (e.g., sizes of loans, years on the job, etc.). To improve the performance of the linear models we tested, we first discretized the continuous variables into four broad categories using sklearn's KBinDiscretizer.

After preparing the data, we assessed the difficulty of the core classification task by using the dataset to train two widely used classifiers and assessing them using standard measures (Precision, Recall, and F1).

4.2. Assessment of Classifier Performance

For this part of the evaluation, we compared Random Forest and XGBoost classifiers. Random Forest forms multiple decision trees during training, using randomly diversified input variables and data instances to enhance the model's generalizability. XGBoost employs a sequential gradient-boosting approach, successively building models that fix the errors of previous ones, reducing the model's bias and improving its accuracy [26].

Implementation of these algorithms, as well as the process of generating different feature combinations, was achieved using Python and the NumPy library for efficient mathematical computations. We further harnessed Python's native functionalities in the post-processing stage to generate contrastive anchors, evaluate changes in predictions, and prune redundant supersets.

4.3. Setup for Machine Learning Models

We initiated our empirical analysis with two machine learning models: Random Forest and XGBoost. These models were evaluated using accuracy, precision, recall, and F1 score, across train-validation-test splits of 80%-10%-10%, 70%-15%-15%, and 60%-20%-20%, as presented in Table 3.

The F1 score of the XGBoost model remained relatively steady across all splits, with a decline at a 70%-15%-15% split. In contrast, the Random Forest model improved its F1 score as the training data proportion increased, achieving the peak score of 0.75 with an 80%-10%-10% split.

Given the best F1 score was achieved by the Random Forest model, configured with an 80%-10%-10% split, this model and split were used for subsequent counterfactual evaluations.

Table 4 delineates the specific parameters used for the Random Forest and XGBoost models in these experiments. The parameter values were selected empirically to optimize the models' performance on the training data, based on our preliminary experiments with the dataset. For the Random Forest model, we used a 'log_loss' criterion and log2 max features with 100 estimators and five jobs. Similarly, the XGBoost model was configured with a 'logloss' evaluation metric and a specific setting for tree depth, subsample by the tree, and the number of estimators.

Table 3. Performance metrics of Random Forest and XGBoost under different data split ratios.

Model	Data Split Ratio	Accuracy	Precision	Recall	F1 Score
Random Forest	80%-10%-10%	0.90	0.92	0.63	0.75
	70%-15%-15%	0.90	0.92	0.58	0.71
	60%-20%-20%	0.91	0.88	0.61	0.72
XGBoost	80%-10%-10%	0.90	0.90	0.61	0.73
	70%-15%-15%	0.89	0.88	0.57	0.69
	60%-20%-20%	0.91	0.85	0.62	0.72

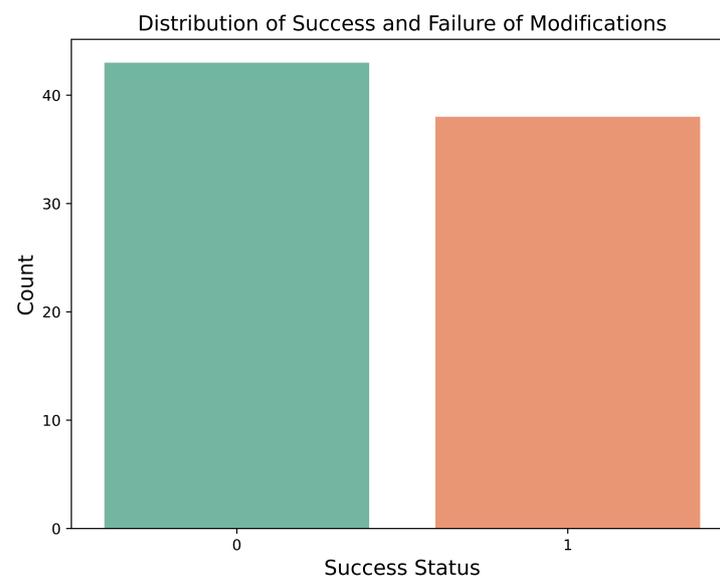
Table 4. Parameters for Random Forest and XGBoost models.

Model	Parameters
Random Forest	n_estimators = 100, n_jobs = 5, criterion = 'log_loss', max_features = 'log2'
XGBoost	eval_metric = 'logloss', max_depth = 6, subsample = 0.8, colsample_bytree = 0.8, n_estimators = 100

4.4. Evaluation of the BFS-AMCC Approach

After selecting the underlying AI model, the BFS-AMCC algorithm was systematically applied to instances initially categorized as 'BAD' by the model. For this analysis, we treated as immutable the indices 4, 5, 6, 7, and 9 (corresponding to REASON, JOB, YOJ, DEROG, and CLAGE, respectively) and specified transition rules to prevent an increase in four variables (DEBTINC, NINQ, DELINQ, and MORTDUE). This resulted in 81 instances from the test set.

Our findings for the success rate of BFS-AMCC showed that, despite the constraints limiting actionability, the algorithm successfully found viable modifications for 38 out of 81 instances, yielding a success rate of 47%. On the other hand, the algorithm failed to find a feasible counterfactual for 43 instances within the set time limit. We excluded the top two outliers from this count, which took an unusually long processing time of 2.75 s (which was about ten times the average). These results, visualized in Figure 1, underscore the efficacy of the BFS-AMCC algorithm in finding potent, achievable modifications, even under the limitations imposed by immutable variables and transition rules.

**Figure 1.** Distribution of success (1) and failure (0) of modifications.

To assess suitability for real-time use, we calculated the time taken for modifications and created a histogram. Figure 2 shows the percentage distribution of the time taken for each modification, showing that more than 90% of modifications were completed in 400 ms or less, regardless of success.

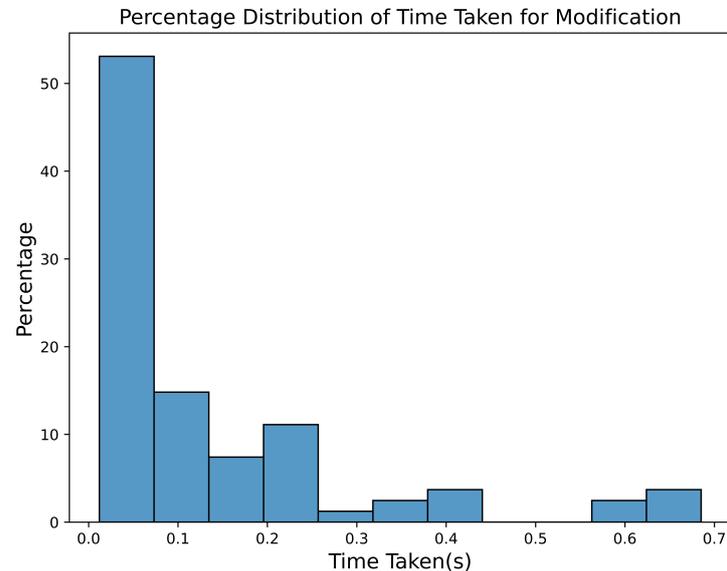


Figure 2. Distribution of time taken (in seconds) for modification.

We also compared the time taken for successful and unsuccessful modifications and constructed a box plot, as shown in Figure 3. The plot shows that for both successful and unsuccessful modifications, the median times for obtaining an answer were well below 200 ms, clearly achieving long-established acceptable latency for interactive tasks on a variety of devices [18,19]. This comparison confirms the suitable efficiency of the BFS-AMCC algorithm for use in real-time interactive systems.

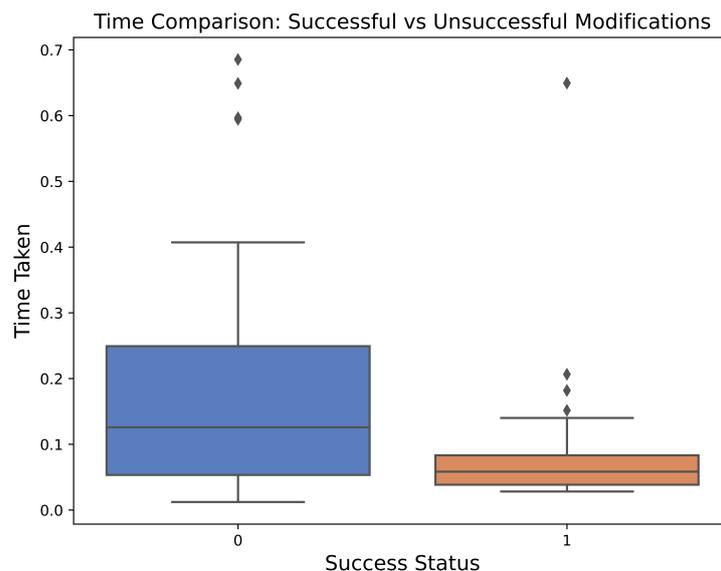


Figure 3. Comparison of time taken for successful (1) vs. unsuccessful (0) modifications.

4.5. Evaluation of Impact of Beam Size

The use of the Anchors algorithm requires specifying ‘beam size’, which is the maximum number of candidates that it will store while searching, where candidates are ordered by how well they cover the data. For the purposes of speed, a smaller beam might seem

better, however, using too small a beam can lead to discarding successful candidates, increasing the length of the search. Since determining which beam size to use depends on the data and how many features are mutable, we tested a range of values for the beam, under two conditions: restrict BFS-AMCC to use only actionable features and any transition rules (Ignore indices are TRUE) or let BFS-AMCC use any anchor (Ignore indices are False).

Table 5 shows the performance of the BFS-AMCC algorithm under three different beam sizes and conditions of ignored indices with transition rules (the typical use case) versus full flexibility. The total time is the aggregate for the test set. (We consider the average for an individual case below). Not surprisingly, the findings show that granting the algorithm complete freedom in feature manipulation, by setting both ignored indices and transition rules to FALSE, bolstered success rates across all examined beam sizes. However, this unconstrained exploration came with a high computational overhead, evidenced by a substantial increase in execution time. On the other hand, enforcing constraints via ignored indices and transition rules (the TRUE condition) resulted in a lower success rate, but yielded a significant reduction in computation time. These observations underscore the importance of careful parameter tuning (for beam size) and placing realistic constraints on feature perturbation in the optimization of BFS-AMCC. There is a delicate trade-off between success rate, computational efficiency, and the extent of allowable feature manipulations that will need to be assessed for each domain.

Table 5. Performance metrics for different beam sizes and ignore indices conditions.

Ignore Indices	Beam Size	Success	Failure	Total Time (s)
TRUE	3	35	46	13.99
	5	37	44	10.87
	10	38	43	11.39
FALSE	3	55	26	213.78
	5	59	22	110.81
	10	59	22	151.1

Table 6, shows the results when calculating the mean time for both successful and unsuccessful modifications. The results indicate that the average time taken for a successful modification is only 80 ms and much shorter compared to unsuccessful ones, 190 ms on average.

Table 6. Mean time for successful and unsuccessful modifications.

Success Status	Mean Time (s)
Successful	0.08
Unsuccessful	0.19

4.6. Examples of BFS-AMCC Modifications that Improve Predicted Outcomes

As a final assessment of the practical utility of the BFS-AMCC approach, we considered which recommendations might be provided, and how long it took to find these explanations, for a sample of test cases. Figure 4 shows a collection of representative instances reflecting variable modifications and their corresponding computational times. All these changes in variable values were performed by the BFS-AMCC algorithm on instances initially classified as 'BAD'. Each row represents a unique instance with modified variable(s) and the time taken. For example, the fourth row illustrates a change in the 'DELINQ' variable from 4.0 to 2.0, corresponding to a reduction of delinquent credit lines from 11 to 15 to the range 4–7. Finding this result took 63 ms. As the figure shows, the computational time typically remained under 500 ms, even when multiple variables were

altered simultaneously. These findings exemplify the efficiency of the BFS-AMCC algorithm in transforming data instances while minimizing computational burden.

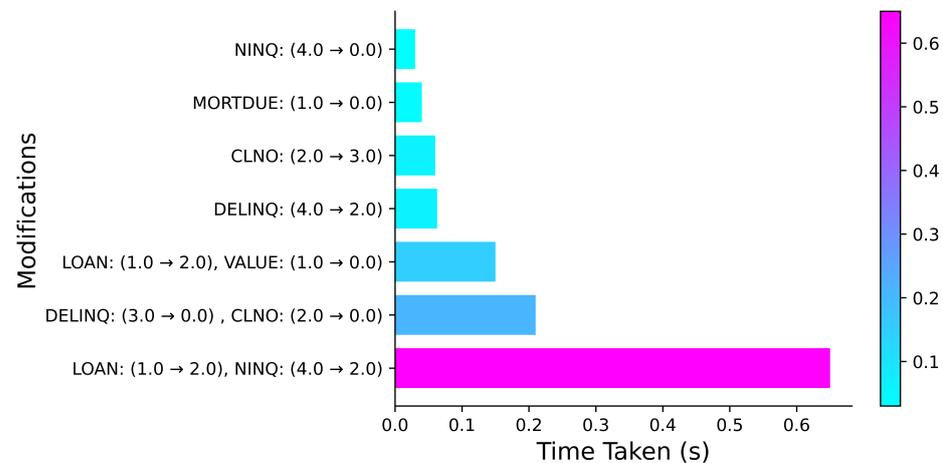


Figure 4. Variable modifications and their respective durations.

5. Discussion

Overall, the findings indicate that one can use BFS-AMCC to provide high-precision explanations that respect the information provided, including which features are actionable and which transitions are achievable. Success rates for finding achievable minimally-contrastive counterfactual explanations were comparable to finding unconstrained high-precision predictions and were found much more quickly. Across all tested beam sizes, the AMCC results took about one-tenth the time for an unconstrained search, and nearly always less than 200 ms. The algorithm found AMCC explanations 47% of the time (38 of 81 test cases) and took on average just 80 ms to find a viable AMCC explanation. Thus, our primary research questions were addressed.

Applying the algorithm to a real-world lending dataset and observing its success in altering numerous ‘BAD’ instances show the practical relevance of BFS-AMCC. For example, both decision-makers and loan applicants might benefit from help identifying requisite changes for improved outcomes or discerning when no feasible recourse exists. These insights could also support educational initiatives aimed at enhancing financial literacy.

As highlighted by the distribution of successful and unsuccessful modifications shown in Figure 1, the BFS-AMCC algorithm effectively modifies a majority of instances, addressing our initial research question, about whether the method can find recourse. Furthermore, these modifications are executed rapidly, typically within 200 ms, responding to our second research query regarding the speed of recourse identification. By integrating a stringent time constraint within the BFS-AMCC, the real-time efficiency of the algorithm can be further enhanced.

Insights from Table 5 illustrate how the performance of the BFS-AMCC algorithm is influenced by variations in beam size and ignore-indices conditions. It appears that an equilibrium between computational resources and algorithm performance needs to be maintained and evaluated empirically. With ignore indices set to TRUE, consistent success rates across various beam sizes were observed, with smaller beam sizes offering enhanced efficiency. Conversely, when ignore-indices were FALSE, a larger beam size led to a minor success rate increase, but a significant rise in computational time. This finding underscores the importance of judiciously choosing the beam size and setting ignore indices, a conclusion further substantiated by the exclusion of top outliers under ignore indices set to FALSE.

The contrast in time taken for successful versus unsuccessful modifications (Figure 3 and Table 6) elucidates another key aspect. Despite initially seeming paradoxical, the faster completion of successful modifications can be explained by the breadth-first search

nature of the BFS-AMCC algorithm, which finds achievable modifications early in the traversal. In contrast, unsuccessful modifications necessitate a more exhaustive search before concluding no successful modification exists. This suggests that in real-time use it is safe to set a time-bound and respond with a negative outcome if it is reached.

Figure 4 showcases the efficiency of the BFS-AMCC algorithm in adjusting data instances. The ability to modify multiple variables simultaneously while maintaining low computational times exemplifies the optimized search and transformation capabilities of the algorithm. This highlights the algorithm's potential for scalability and suitability in complex scenarios. Furthermore, the successful execution of transformations within certain constraints, such as immutable variables, illustrates its adaptability and versatility, broadening its potential application across diverse problem-solving domains.

Finally, the superior performance of the Random Forest classifier in our experiments emphasizes the significance of model selection in such analyses. The classifier's ability to navigate high-dimensional feature spaces and intricate decision boundaries in the dataset may account for its success. This study offers insights into the interaction between different models and the BFS-AMCC procedure, providing a foundation for future research. For example, for datasets with large numbers of features, one might want to consider automated feature selection.

5.1. Limitations

Despite its merits, the BFS-AMCC methodology exhibits several limitations that warrant mention. The algorithm allows the inclusion of immutable features using the 'ignore indices' parameter. However, discerning mutable features and their respective extents can be challenging in real-world contexts, necessitating domain-specific expertise. Deciding an appropriate time boundary, especially in scenarios with no evident feasible recourse, remains an avenue for future exploration. Furthermore, systematic evaluations across various parameter settings are crucial to ensure the robustness of machine learning models.

A meticulous analysis of the HMEQ dataset underscores that the notion of achievability needs to be intrinsically tied to individual actions and transitions, crafted in line with domain-specific constraints. For instance, it is logically plausible to reduce one's mortgage through advanced payments, but inversely, it is not feasible to augment it. When a model suggests that such an increment leads to more favorable outcomes, it indirectly alludes to the inherent advantages of initial affluence. In our methodology, this domain nuance is managed by implementing specific transition rules, one of which is set to negate any increments in the 'MORTDUE' variable. However, as portrayed in Figure 4, certain allowed alterations might demand direct user interactions. An illustrative case is the change in CLNO (2 → 3), which implies an increment in open credit lines from the bracket of 18–35 to 36–53. The viability of such adjustments largely pivots on individual situations and potential intricacies between variables not considered in our analysis.

It is also worth noting that while specific modifications might accurately mirror the dataset's directives and could be apt for recommendations, their practicality could waver based on the specific context of the individual. An example is the DELINQ transition (4.0 → 2.0), which suggests trimming the number of delinquent credit lines from a maximum of 15 to fewer than seven. The feasibility of such advice is inherently tethered to numerous factors, including debt magnitudes. An interactive approach would enable users to counter such propositions, prompting the AMCC to subsequently treat aspects such as DELINQ as immutable. This dynamic nature also allows for addressing specific queries, such as the 'what-about-X' style, by making only the parameter X mutable. Future work might explore adapting the AMCC approach to employ a beam search over the traditional breadth-first, yielding a gamut of counterfactual candidates. These alternatives can then be sequenced based on individual-centric utilities or likelihoods of success. This conceptualization aligns with prior research in counterfactuals that emphasizes presenting users with a diverse set of interpretative alternatives to ensure a holistic understanding.

5.2. Comparative Analysis with Previous Studies

The proposed methodology attains the objective of identifying achievable minimal counterfactuals without necessitating the computation of intricate metrics for optimization. In alignment with Poyiadzi et al.'s FACE approach [23], the presented algorithm encompasses feasibility and actionability in a model-agnostic fashion. Contrarily, the AMCC procedure, distinct from FACE, does not determine feasibility grounded on the density of dataset examples and refrains from calculating similarity metrics. Both FACE and AMCC deploy search mechanisms for counterfactual identification, with both breadth-first search and Dijkstra's algorithm targeting the shortest paths. While BFS employs a rudimentary queue, Dijkstra's algorithm leverages weighted edges paired with a priority queue. Noteworthy, the proposed method does not mandatorily pre-construct a graph prior to search initiation; it instead incrementally alters instances feature-wise, concomitantly and implicitly instantiating a new node, each epitomizing a minimal alteration of its predecessor. Furthermore, the assessment herein incorporates pragmatic metrics, including the time duration for counterfactual discovery, an aspect overlooked in previous research.

Mothilal et al. [25], paralleling Poyiadzi et al., conceptualize the derivation of minimal achievable counterfactuals as an optimization task, thereby necessitating proximity and diversity calculations. Their evaluative criteria span F1, Recall, and Precision, measured across diverse datasets, albeit excluding time considerations. Encompassing all datasets and methods under scrutiny, their technique registers an F1 in the range of 0.2 to 0.6, similar to the observed mean success rate of 47% for AMCC in this study.

5.3. Future Research

Future research can explore in depth the influence of different predictive models on the performance of the BFS-AMCC procedure, investigate its adaptability to data with varying degrees of feature mutability, and assess potential alternative strategies for counterfactual generation that might offer equal or superior efficacy. An intriguing avenue is the extraction of domain-specific information required by the algorithm from more generalized representations, such as knowledge graphs [27]. Such endeavors would further the application of AMCC in the development of conversational agents designed to encourage sustained behavioral modifications, including enhancements in time management or augmentation of physical activity [28,29]. This direction aligns with the trajectory of our ongoing research.

Despite the inherent intricacies, the insights presented herein signify a pivotal progression in the comprehension and application of counterfactuals for enhancing the interpretability of machine learning in both prediction and intervention, laying the groundwork for subsequent advancements.

6. Conclusions

In this paper, the BFS-AMCC algorithm is presented as a robust method to identify achievable minimally-contrastive counterfactuals compatible with any machine learning framework catering to tabular data. The provision of such high-precision, model-agnostic counterfactuals is essential for decision support systems powered by machine learning, emphasizing the pivot from mere prediction to actionable intervention. In interactive scenarios, when users confront a negative outcome, they seek elucidation on its causes and viable avenues to a favorable result. Ideally, a system should respond with actionable insights within a latency of 200 to 500 ms, a timeframe deemed acceptable by prior research in human-computer interaction. Notably, this method forgoes the dependency on intricate metrics to gauge feasibility. Instead, it leans on precise directives from domain experts to delineate feasible actions, with the system enumerating alternatives within a predetermined timeframe. Potential unforeseen causal discrepancies or conflicts are addressed via user interaction, yielding a refined AMCC explanation. Therefore, this paradigm for explainable AI is inherently transparent and decipherable to a wide spectrum of individuals. The real-time efficacy of the AMCC strategy underlines its potential in powering conversational interfaces aimed at guiding users toward enhanced results.

Overall, the findings presented here confirm that our BFS-AMCC algorithm usually generates at least one actionable explanation that permits meaningful intervention, all within an acceptable time frame. This study, therefore, provides compelling evidence for the efficacy of BFS-AMCC. Moreover, the BFS-AMCC approach meets key requirements for real-world applications and end-user interpretability, underscoring its practical potential.

Author Contributions: Conceptualization, H.B. and S.M.; methodology, H.B. and S.M.; software, H.B.; validation, H.B. and S.M.; formal analysis, H.B.; investigation, H.B.; resources, H.B. and S.M.; data curation, H.B.; writing—original draft preparation, H.B. and S.M.; writing—review and editing, S.M.; visualization, H.B.; supervision, S.M.; project administration, S.M. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge the support of the Office of Dean of the College of Engineering & Applied Science at the University of Wisconsin-Milwaukee.

Data Availability Statement: The data from this study is derived from the widely available Home Equity Loan Default dataset. The original dataset can be found on Kaggle at: <https://www.kaggle.com/datasets/lchipham/home-equity-loan-default>, <https://www.kaggle.com/datasets/ajay1735/hmeq-data> (accessed on 11 May 2023). For reproducibility and further exploration, our code and modified data are online and public. Access our repository on GitHub at: https://github.com/barzekar/BFS_AMCC_XAI.

Acknowledgments: The authors thank their families for their continued patience and support.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AMCC	Achievable Minimally-Contrastive Counterfactual Explanation
AI	Artificial Intelligence
BFS	Breadth-First Search
CCE	Contrastive Counterfactual Explanation
RF	Random Forest
XAI	Explainable Artificial Intelligence

References

- Ribeiro, M.T.; Singh, S.; Guestrin, C. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'16, San Francisco, CA, USA, 13–17 August 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 1135–1144. [CrossRef]
- Shin, D. The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable AI. *Int. J. Hum.-Comput. Stud.* **2021**, *146*, 102551.
- Ribeiro, M.T.; Singh, S.; Guestrin, C. Anchors: High-Precision Model-Agnostic Explanations. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), New Orleans, LA, USA, 2–7 February 2018.
- Deci, E.L.; Ryan, R.M. The "what" and "why" of goal pursuits: Human needs and the self-determination of behavior. *Psychol. Inq.* **2000**, *11*, 227–268.
- Ryan, R.M.; Deci, E.L. Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *Am. Psychol.* **2000**, *55*, 68–78. [PubMed]
- Ryan, R.M.; Deci, E.L. Brick by brick: The origins, development, and future of self-determination theory. In *Advances in Motivation Science*; Elsevier: Amsterdam, The Netherlands, 2019; Volume 6, pp. 111–156.
- Kwasnicka, D.; Dombrowski, S.U.; White, M.; Sniehotka, F. Theoretical explanations for maintenance of behaviour change: A systematic review of behaviour theories. *Health Psychol. Rev.* **2016**, *10*, 277–296. [PubMed]
- Doran, G.T. There's a SMART way to write management's goals and objectives. *Manag. Rev.* **1981**, *70*, 35–36.
- Burik, A. Using Technology to Help Students Set, Monitor, and Achieve Goals. *Adult Lit. Educ.* **2021**, *3*, 83–89.
- Gupta, A.; Basu, D.; Ghantasala, R.; Qiu, S.; Gadiraju, U. To trust or not to trust: How a conversational interface affects trust in a decision support system. In Proceedings of the ACM Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 3531–3540.
- Mittelstadt, B.; Russell, C.; Wachter, S. Explaining explanations in AI. In Proceedings of the Conference on Fairness, Accountability, and Transparency, Atlanta, GA, USA, 29–31 January 2019; pp. 279–288.

12. Kuźba, M.; Biecek, P. What would you ask the machine learning model? Identification of user needs for model explanations based on human-model conversations. In Proceedings of the ECML PKDD 2020 Workshops: Workshops of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, 14–18 September 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 447–459.
13. Wachter, S.; Mittelstadt, B.; Russell, C. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL Tech.* **2017**, *31*, 841.
14. Stepin, I.; Alonso, J.M.; Catala, A.; Pereira-Fariña, M. A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence. *IEEE Access* **2021**, *9*, 11974–12001.
15. Karimi, A.H.; Schölkopf, B.; Valera, I. Algorithmic recourse: From counterfactual explanations to interventions. In Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency, Toronto, ON, Canada, 3–10 March 2021; ACM: New York, NY, USA, 2021; pp. 353–362.
16. Karimi, A.H.; Von Kügelgen, J.; Schölkopf, B.; Valera, I. Algorithmic recourse under imperfect causal knowledge: A probabilistic approach. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 265–277.
17. Kaddour, J.; Lynch, A.; Liu, Q.; Kusner, M.J.; Silva, R. Causal Machine Learning: A Survey and Open Problems. *arXiv* **2022**, arXiv:cs.LG/2206.15475.
18. Dabrowski, J.; Munson, E.V. 40 years of searching for the best computer system response time. *Interact. Comput.* **2011**, *23*, 555–564. [[CrossRef](#)]
19. Yu, M.; Zhou, R.; Cai, Z.; Tan, C.W.; Wang, H. Unravelling the relationship between response time and user experience in mobile applications. *Internet Res.* **2020**, *30*, 1353–1382.
20. Lundberg, S.M.; Lee, S.I. A unified approach to interpreting model predictions. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
21. Rath, S. Generating counterfactual and contrastive explanations using SHAP. *arXiv* **2019**, arXiv:1906.09293.
22. Pearl, J. Structural counterfactuals: A brief introduction. *Cogn. Sci.* **2013**, *37*, 977–985. [[CrossRef](#)]
23. Poyiadzi, R.; Sokol, K.; Santos-Rodriguez, R.; De Bie, T.; Flach, P. FACE: Feasible and actionable counterfactual explanations. In Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, 7–8 February 2020; pp. 344–350.
24. Ustun, B.; Spangher, A.; Liu, Y. Actionable recourse in linear classification. In Proceedings of the Conference on Fairness, Accountability, and Transparency, Atlanta, GA, USA, 29–31 January 2019; pp. 10–19.
25. Mothilal, R.K.; Sharma, A.; Tan, C. Explaining machine learning classifiers through diverse counterfactual explanations. In Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency, Barcelona, Spain, 27–30 January 2020; pp. 607–617.
26. Murphy, K.P. *Probabilistic Machine Learning: An Introduction*; MIT Press: Cambridge, MA, USA, 2022.
27. Hogan, A.; Blomqvist, E.; Cochez, M.; d’Amato, C.; Melo, G.D.; Gutierrez, C.; Kirrane, S.; Gayo, J.E.L.; Navigli, R.; Neumaier, S.; et al. Knowledge graphs. *ACM Comput. Surv. (Csur)* **2021**, *54*, 1–37. [[CrossRef](#)]
28. Kimani, E.; Rowan, K.; McDuff, D.; Czerwinski, M.; Mark, G. A conversational agent in support of productivity and wellbeing at work. In Proceedings of the 2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII), Cambridge, UK, 3–6 September 2019; pp. 1–7.
29. Zhang, J.; Oh, Y.J.; Lange, P.; Yu, Z.; Fukuoka, Y. Artificial intelligence chatbot behavior change model for designing artificial intelligence chatbots to promote physical activity and a healthy diet. *J. Med. Internet Res.* **2020**, *22*, e22845. [[PubMed](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.