Survey of Explainable AI: Interpretability and Causal Reasoning

Prashant K. Sharma
IIT Bombay
prashaantsharmaa@gmail.com

Pushpak Bhattacharyya
IIT Bombay
pb@cse.iitb.ac.in

Abstract

Humans do have the ability to explain the rationale or process behind the decision they make: be it intuition, observation , experience or logical thinking. Coming to complex AI algorithms e.g deep learning, they are treated as black-box. Thus, as these algorithms become more and more pervasive into human lives, the need for explainable artificial intelligence arises. This research seminar focuses on understanding various aspects of this field, the various approaches that have been proposed in this domain and it's application to the field of Natural Language Processing. The aim has been to come up with key questions to be asked in this domain which can be further investigated and used to develop state-of-the art models in Natural Language Processing, keeping in mind performance with explainability component. In this survey of the field we will look at Interpretability algorithms and approach to causal reasoning systems.

1 Introduction

The models that are trained on large amounts of data, that is mostly created by users. Thus, likely to contain human biases and prejudices.Therefore, models that are learned i.e trained using these data will also carry such prejudices and biases which can be harmful.

1.1 Need for Interpretable Models

Training a classifier on historical datasets, reporting human decisions, could lead to the discovery of some bad preconceptions. Moreover, since these rules can be deeply concealed within the trained classifier, we risk to consider, maybe unconsciously, such practices and prejudices as general rules. We are warned about a growing black box society, governed by secret algorithms protected by industrial secrecy, legal protections, obfuscation, so that intentional or unin- tentional discrimination becomes invisible and mitigation becomes impossible. Automated discrimination is not new and is not necessarily due to black box models. A computer program for screening job applicants were used during the 1970s and 1980s in St. Georges Hospital Medical School (London). The program used information from applicants forms, without any reference to eth- nicity. However, the program was found to unfairly discriminate against ethnic minorities and women by inferring the information from surnames and place of birth, and lowering their chances of being selected for interview.

A study at Princeton University shows how text and web corpora can contain human biases: names associated with the black people are noted to be significantly more associated with unpleasant terms than that of pleasant terms, in comparison to the names of whiltes. Therefore, the models that are learned on such text data have a huge possibility of having the prejudices reflected in the data.

If we look at another example that is related to Amazon.com. In the year 2016, the software used by Amazon that was used to determine the regions of the US to which Amazon would offer free same-day delivery, unintentionally restricted minority regions from participation into the program (often when the surrounding regions were allowed.
There are many works where it is shown that how accurate black box classifiers can result from an

accidental pertubation in the training data. For example, US military trained a deep learning classifier to distinguish between enemy tanks and friendly tanks. The classifier gave high accuracy on test set, but performed poorly on real data. It was later discovered that enemy tank photos were taken on overcast days, while friendly tank photos on sunny days. Similarly, it is also shown that a classifier trained to recognize wolves and husky dogs were basing its predictions to classify a wolf solely on the presence of snow in the background.
Nowadays, Deep Neural Networks (DNNs) have been reaching very good per- formances on different pattern-recognition tasks such as visual and text clas- sification which are easily performed by humans: e.g., saying that a tomato is displaced in a picture or that a text is about a certain topic. Thus, what differences remain between DNNs and humans? Despite the excellent performance of DNNs it seems to be a lot. It is shown the alteration of an image (e.g. of a tomato) such that the change is undetectable for humans can lead a DNN to tag the image as something else (e.g., labeling a tomato as a dog). It is also shown that it is easy to produce images that DNNs believe to be recognizable with 99.99 % confidence, but which are completely unrecognizable to humans (e.g., labeling white static noise as a tomato).

2 Explainability of Deep Neural Networks

There are two aspects when it comes to explanations of the operation of deep networks: one is explaining the processing of the data by the network, another is the representation of the data inside the network. This explanation of processing asks the question "Why this input leads to that output?". Now, explanation about representation asks "What information is contained by the network?".

2.1 Explanations of Deep Network Representations

Although the number of individual operations in a network is huge, deep networks can be organized into a much smaller number of sub components: example: The aim with explanation of deep network representations is to understand the role and associated structure of the data that flows through these bottlenecks. This work can be divided as follows :

1. **Role of Layers**:The property of layers can be understood if they show the ability to help solve various problems from the whatever the network was originally trained on. Ref: [Sharif Razavian et al.2014]

2. **Role of Individual Units**: When we look at a layer: a layer can be subdivided into individual neurons or individual convolutional filters. The role of such individual units can be understood:

   - qualitatively: by creating visualizations of the input patterns that maximize the response of a single unit
   - quantitatively, by testing the ability of a unit to solve a transfer problem

2.2 Algorithms for Explanability

2.2.1 Sensitivity Analysis

In this section, we describe another strategy which is is inspired by the back-propagation strategy in vision [4]. It measures how much each input unit contributes to the final decision, which can be approximated by first derivatives.

In the case of deep neural models, the class score $S_c(e)$ is a highly non-linear function. We approximate $S_c(e)$ with a linear function of e by computing the first-order Taylor expansion

$$S_c(e) = w(e)^T e + b$$

where $w(e)$ is the derivative of S_c with respect to the embedding e.

$$w(e) = \frac{\partial S_c}{\partial e}$$

The saliency score is given by

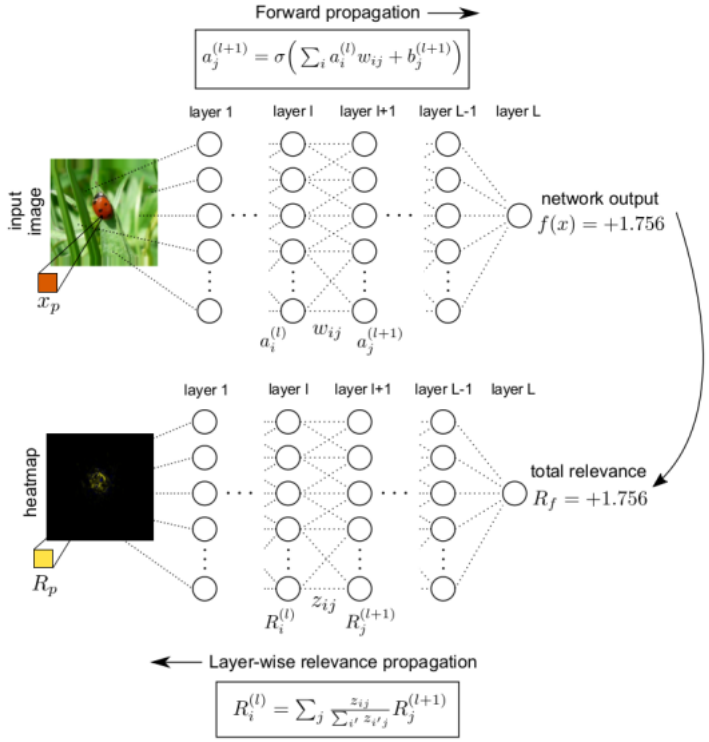

Figure 1: Layer wise Relevance Propagation [1]

$$S(e) = |w(e)|$$

Understanding Sensitivity Analysis Sensitivity analysis explains variation of the function not the function value itself.
Observation:

$$\sum_{i=1}^{d} (\frac{\partial f}{\partial x_i}) = ||\nabla_x f||^2 \tag{1}$$

Problem: Shattered gradients. Input gradient (on which sensitivity analysis is based), becomes increasingly highly varying and unreliable with neural network depth.

2.2.2 Layerwise Relevance Propagation

Layerwise Relevance Propagation(LRP) is not sensitive to gradients unlike sensitivity analysis. A deep neural network is a feed-forward graph of elementary computational units (neurons), each of them realizing a simple function. The same graph structure can be used to redistribute the relevance $f(x)$ at the output of the (1) network onto pixel-wise relevance scores R_p , by using a local redistribution rule as follows:

$$R_i^{(l)} = \sum_j \frac{z_{ij}}{\sum_{i'} z_{i'j}} R_j^{(l+1)} \quad with \quad \mathrm{z_{ij}} = \mathrm{x_i^{(l)}} \mathrm{w_{ij}^{(l,1+1)}} \tag{2}$$

Application of this rule in a backward pass produces a relevance map (heatmap) that satisfies the desired conservation property : the relevance that flows in a layer of neural network, same flows out from that layer. This decom- position algorithm is termed Layer-wise Relevance Propagation (LRP).

2.2.3 LIME: Locally Interpretable Model-agnostic Explanations

This section is based on [Ribeiro et al.2016] The goal of LIME is to identify an interpretable model over the interpretable representation that is locally faithful to the classifier.
Intuition

- Perturb the input and see how the predictions change. In terms of interpretability, change the components that make sense to humans (e.g. words or part of an image), even if model is using complicated components as features (eg. word embeddings).
- **Key intuition**: Easier to approximate a black-box model by a simple model locally (in neighborhood of the prediction we want to explain) as opposed to trying to approximate a model globally.

2.3 Integrated Gradients

Introduced in Axiomatic Attribution for Deep Networks by [Sundararajan et al.2017].
Problem Statement: Attribute a deep network's prediction to its input features
E.g. Attribute a diagnostic network's prediction to patient's symptoms, measurements, history
E.g. Attribute an Object Recognition Network's prediction to its pixels

2.3.1 Approach: Axioms

- We list desirable criteria (axioms) for an attribution method
- Integrated Gradients the only method that satisfies these desirable criteria.

2.3.2 Baselines: Need

- A baseline is an uninformative input used for comparison
  - Object recognition: black image, noise image
  - For f = x*y + z, and attribution at x,y,z=1,1,1, natural baseline is 0,0,0
- Baselines are necessary for attributions

2.3.3 Axiom: Implementation Invariance

Two networks that compute identical functions for all inputs get identical attributions even if their architecture/parameters differ
E.g. f1 = x*y + z and f2 = y*x + z should get the same attributions
Not satisfied by Layer-wise relevance propagation

2.3.4 Axiom: Sensitivity

- If baseline and input have different scores, but differ in a single variable,then that variable gets some attribution.
- If a variable has no influence on a function, then it gets no attribution.

E.g. f1 = x*y + z and f2 = y*x + z should get the same attributions
Not satisfied by Layer-wise relevance propagation

2.3.5 Axiom: Linearity Preservation

- Attributions(*f1 + ß*f2) = *Attributions(f1) + ß*Attributions(f2)
- Attributions have additive semantics. It is best to respect any existing linear structure.

[2] http://www.heatmapping.org/

2.3.6 Axiom: Completeness

- sum(attributions) = f(input) - f(baseline)
- Every method that satisfies Linearity preservation, Sensitivity and Implementation invariance, and Completeness is a path integral of a gradient.

2.3.7 Axiom: Symmetry Preservation

Symmetric variables with identical values get equal attributions.

- E.g. For f = x*y + z, the "optimal" attribution at x,y,z=1,1,2 should be equal for x and y.
- Integrated Gradients is the unique path method that satisfies these axioms. (there are other methods that take an average over a symmetric set of paths)

2.4 Using influence functions to understand influence

This section is based on [Koh and Liang2017].

GOAL:Trace a model's prediction back to its training data, identifying training points most responsible for given prediction.
Approach: Two ways

- Upweighting a training point
- Perturbing a training input

Calculationg Influence:

- Computational challenges
- Conjugate gradents(CG)
- Stochastic Estimation

2.4.1 Upweighting a training point

- How would the model's predictions change if we did not have this training points ?
- Naive approach: remove the training point and retrain the network (slow)
- Influence functions: give us an approximations
- IDEA: compute the parameter change if z were upweighted by small ϵ

$$\hat{\theta}_{\epsilon,z} \overset{def}{=} \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} L\left(z_i,\theta\right) + \epsilon L(z,\theta) \tag{3}$$

- Influence function is given by

$$\mathcal{I}_{up,params}(z) \overset{def}{=} \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1}\nabla_\theta L(z,\hat{\theta}) \tag{4}$$

  where,

$$H_{\hat{\theta}} \overset{\text{def}}{=} \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta^2 L\left(z_i,\hat{\theta}\right) \tag{5}$$

  Which is the hessain and is positive defininte(PD) by assumptions

- The influence of upweighting z on the loss at the test point z_{test} has closed form expression

$$\mathcal{I}_{uploss}\left(z,z_{test}\right) = \left.\frac{dL\left(z_{test}\hat{\theta}_{\epsilon,z}\right)}{d\epsilon}\right|_{\epsilon=0} = \nabla_\theta L\left(z_{test},\hat{\theta}\right)^\top \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -\nabla_\theta L\left(z_{test},\hat{\theta}\right)^\top H_{\hat{a}}^{-1}\nabla_\theta L(z,\hat{\theta}) \tag{6}$$

2.4.2 Perturbing a training point

- Consider perturbation $z \rightarrow z_\delta$

$$\left.\frac{d\hat{\theta}_{\epsilon,z_\delta,-z}}{d\epsilon}\right|_{\epsilon=0} = \mathcal{I}_{\text{up},params}\left(z_\delta\right) - \mathcal{I}_{up,params}(z) = -H_{\hat{\theta}}^{-1}\left(\nabla_\theta L\left(z_\delta,\hat{\theta}\right) - \nabla_\theta L(z,\hat{\theta})\right) \quad (7)$$

- Approximating the perturbation effect on the loss at z_{test}

$$\mathcal{I}_{pertloss}\left(z,z_{test}\right)^\top \stackrel{def}{=} \nabla_\delta L\left(z_{test},\hat{\theta}_{z_\delta,-z}\right)^\top\Big|_{\delta=0} = -\nabla_\theta L\left(z_{test},\hat{\theta}\right)^\top H_{\hat{\theta}}^{-1}\nabla_x\nabla_\theta L(z,\hat{\theta}) \quad (8)$$

2.4.3 Efficiently calculating influence

Two challenges in calculating equation 8

- Requires forming and inverting hessian of empirical risk, with training examples its expensive for deep neural networks
- We would like to calculating equation8 for all training points

2.5 Semantically Equivalent Rules for Adversarial Examples

This section is based on [Ribeiro et al.2018]

2.5.1 Overview

SEARs: Semantically equivalent adversaries Rules

- Proposing a set of Rules
- Selecting a set of Rules
  - Semantic Equivalence
  - High Adversary Count
  - Non-redundancy
- Generating SEARs for a model

Semantic Score:

$$S\left(x,x'\right) = \min\left(1,\frac{P\left(x'|x\right)}{P(x|x)}\right) \quad (9)$$

Ratio between probability of a paraphrase and the probability of the sentence itself.

$$SemEq\left(x,x'\right) = 1\left[S\left(x,x'\right) \geq \tau\right] \quad (10)$$

$$SEA\left(x,x'\right) = 1\left[SemEq\left(x,x'\right) \wedge f(x) \neq f\left(x'\right)\right] \quad (11)$$

2.6 Sanity check for saliency maps

This section is based on [Adebayo et al.2018]

2.6.1 Overview

- **GOAL**: An actionable methodology based on randomization tests to evaluate the adequacy of explanation approaches.
- Shown that some existing saliency methods are independent both of the model and of the data generating process
- **Claim**:
  - Methodology applies in generality to an explanations approx.
  - Suitability of an explanation method for a given task at hand.
- **Approach**
  - Model parameter randomization test
    - Cascading Randomization
    - Independent Randomization
  - Data randomization test

2.6.2 Model Parameter Randomization Test

- Cascading Randomization
  - **Overview**: Randomize the weights of a models starting from the top layer ,all the way to the bottom layer.
  - **Result**:
    - The gradient map and GradCAM marks are sensitive to model parameters.
    - Guided BackProp and Guided GradCAM - show not change regardless of model degradation - across all architectures and datasets.
- Independent Randomization
  - **Overview**:
    - Layer-by-layer randomization with the goal of isolating the dependence of explanations by layer.
    - Assess the dependence of saliency masks on lower vs. higher layer weights.
  - **Result**: Guided BackProp and Guided GradCAM - show not change

2.6.3 Dataset Randomization Test

By randomizing the labels, evaluates the sensitivity of an explanation method to the relationship between instances and labels.

2.7 Attention is not explanation

This section is based on [Jain and Wallace2019].

2.7.1 Correlation Between Attention and Feature Importance Measures

Empirically characterized the correlation between weights and corresponding features importance scores in two ways. Correlation between attention weights and

1. Gradient based measures of feature importance
2. Difference in model output induced by leaving features out

Adversarial Attention The proposal is the following optimization problem to identify adversarial attention weights.

$$\begin{array}{ll} \alpha^{(1)}, \ldots, \alpha^{(k)} maximize & f\left(\left\{\alpha^{(i)}\right\}_{i=1}^{k}\right) \\ subject to & \forall i TVD\left[\hat{y}\left(\mathbf{x}, \alpha^{(i)}\right), \hat{y}(\mathbf{x}, \hat{\alpha})\right] \leq \epsilon \end{array} \tag{12}$$

$$\begin{array}{l} Where f\left(\left\{\alpha^{(i)}\right\}_{i=1}^{k}\right) is: \\ \sum_{i=1}^{k} JSD\left[\alpha^{(i)}, \hat{\alpha}\right] + \frac{1}{k(k-1)} \sum_{i<j} JSD\left[\alpha^{(i)}, \alpha^{(j)}\right] \end{array} \tag{13}$$

3 Causality and Causal Reasoning

3.1 The Predicate Calculus in AI

Predicate calculus or simply put logic cab be thought of as a notation for internal representations useful for the database in a production system.

Its characteristics are:predicate calculus allows the deduction of new facts that is based on form or facts, also helps in question answering and planning support.

Logic cannot be thought of as mere representation but a language. We still need to decide what and how of the representation

Logic can also be define as formalism to express something which is true and false. Which can be used to infer new facts. Which helps in our understanding.

What is a proposition ?: Proposition is a statements that cab be eigher TRUE or FALSE but not both at the same time.

Examples:

Predicate:English-meaning

- RAIN:"It is raining heavilty"
- SUNNY:"It is sunny morning"
- MANSOCRATES:"Socrates was a man of great caliber"
- MANTURING:"Turing was a man of great caliber"
- ANHJKFG"I like to go to the movies friday night"

3.2 Propositional Logic

To form propositinoal logi, propositions can be combined with connectives to form sentences.

- **AND**: if both X and Y are true, then (X AND Y) is true
- **OR**: if at least one of X or Y is true then (X OR Y) is true.
- **IMPLIES**: when X IMPLIES Y is TRUE if X is true, then Y is true.
- **NOT**: NOT X assumes the opposite of values of X. That is if X is false,NOT X is true and vice-versa.
- **EQUIV**: if X and Y are both true or both false, then X EQUIV Y is true .

3.3 Truth tables

To tell if a statement is true or false?, we Use a truth table:

X	Y	X AND Y	X OR Y	X IMPLIES Y	NOT X	EQUIV X Y
T	T	T	T	T	F	T
T	F	F	T	F	F	F
F	T	F	T	T	T	F
F	F	F	F	T	T	T

Figure 2: Truth table [3]

3.4 WFF: Well formed formula

A term can be

1. a constant whose value is fixed is a term
2. a variable whose value varies is a term
3. If function f takes n-input, and $i_1, .., i_n$ are said to be the terms, then $f(i_1, ..., i_n)$ is also a term.

An atom can be defined asa predicate with terms for arguments, e.g. $P(t_1, ..., t_n)$:
A *well-formed formula (wff)* is: an atom is a wff.
If A and B are formulas , then

- NOT A
- A OR B
- A AND B
- A IMLIES B
- A Equiv B are all wff.

If A is a wff and B is a free variable in A, then (FOR ALL B) A and (EXISTS B) A are formulas.

3.5 First Order Predicate Calculus: Semantics

Interpretation of the formulas cab be defined as establishing a correspondence between constants, functions, predicates.
More formally, the interpretation of a formula X: a nonempty domain D and assignment of ”values” to constance, symbol and predicate:

- each constant is assigned element of domain D
- each function with n-place input, a mapping from $D^n$to D is assigned.
- each predicate symbol of n-place, a mapping from D^n to T, F is assigned.

SEMANTICS can be defined as : A formula is TRUE under an interpretation if.

- If G and H are evaluated, then NOT G,G AND H, G IMPLIES H, G OR H, etc. have values in the obvious way.
- (FORALL X) G is T if G is T for every x in D,otherwise F
- (EXISTS X) G is T if ONE x in D makes it T else F

3.6 Model Semantics

A MODEL of the domain is:an interpretation that makes our sentences TRUE.

- A TAUTOLOGY (or simply, VALID formula)is a formula that is TRUE under ANY interpretation
- A FALLACY (or, an INCONSISTENT formula)is a formula that is FALSE under ANY interpretation
- A formula is SATISFIABLE if there is at least ONE interpretation that makes it TRUE.

3.6.1 Inference

A formula G is a LOGICAL CONSEQUENCE of a set of formulas $F_1, F_2, ...F_N \; iff$ any interpretation that makes $F_1 AND F_2 AND ... AND F_N$ true ALSO makes G true.

DEDUCTION THEOREM
G is a logical consequence of a set of formulas $F_1 AND F_2 AND ... F_n \quad iff$ $[(F_1 AND F_2 AND ... F_n) IMPLIES G]$ is a tautology.
REFUTATION PROOF
iff $F_1 AND F_2 AND ... F_n AND (NOT G)$ is a fallacy (or inconsistent).

3.6.2 Resolution

Resolution is an inference rule that is both SOUND and COMPLETE.

- SOUND = only true facts (logical consequences) are inferred
- COMPLETE = ALL facts that follow CAN be inferred

.

It is important to note that: Modus Ponens is sound but not complete -I can't infer everything with modus ponens.So usually, to get completeness, we have a collection of inference rules.With resolution, we only need ONE.

The Resolution Principle We have just seen the following inference rule in action. Given two sentences in clause form:
If one clause contains **P** and the other **NOT P**,remove these from the two clauses and form the disjunction of the remaining literals

3.6.3 Forward Chaining

Forward chaining also called forward reasoning is a method of reasoning when we are using an inference engine. Formally, this can be described logically as repeated application of modus ponens.
Application Forward chaining is one of the popular implementation strategy when it comes for expert systems, business and production rule systems. The opposite of forward chaining is backward chaining.

Overview of Methodology Forward chaining starts with the available data and uses inference rules to extract more data (from an end user, for example) until a goal is reached. An inference engine using forward chaining searches the inference rules until it finds one where the antecedent (If clause) is known to be true. When such a rule is found, the engine can conclude, or infer, the consequent (denoted by Then clause), resulting in the addition of new information to its data.

The way how Inference engines work is that it will iterate through this process until a specific goal is reached.

3.6.4 Illustrated Example

Suppose that the goal is to conclude the color of a pet named Fritz, given that he croaks and eats flies, and that the rule base contains the following four rules:

Suppose that the goal is to conclude the color of a pet named Fritz, given that he croaks and eats flies, and that the rule base contains the following four rules:

1. If X croaks and X eats flies - Then X is a frog
2. If X chirps and X sings - Then X is a canary
3. If X is a frog - Then X is green
4. If X is a canary - Then X is yellow

Let us illustrate forward chaining by following the pattern of a computer as it evaluates the rules. Assume the following facts:

- Fritz croaks
- Fritz eats flies

By applying forward reasoning, it can derived that Fritz is green by the inference engine in a series of steps, let's have a look:

1. Since the base facts indicate that "Fritz croaks" and "Fritz eats flies", the antecedent of rule 1 is satisfied by substituting Fritz for X, and the inference engine concludes:
   Fritz is a frog
2. The antecedent of rule 3 is then satisfied by substituting Fritz for X, and the inference engine concludes:
   Fritz is green

3.6.5 Backward Chaining

Backward chaining (or backward reasoning) is an inference method described colloquially as working backward from the goal. It is used in automated theorem provers, inference engines, proof assistants, and other artificial intelligence applications.

3.7 Causal Reasoning for Explainable Artificial Intelligence

3.7.1 ATOMIC: An Atlas of Machine Commonsense for If-Then reasoning

This section is based on [Sap et al.2019]

Overview ATOMIC is a knowledge graph for machine commonsense covering if-then inferential knowledge around everyday situation. It key properties are:

- ATOMIC, a new resource for machine commonsense: 880k if-event-then-* knowledge triples
- Free-form for efficient knowledge gathering
- Models can learn to make commonsense inferences.

ATOMIC dataset

Teaching neural models cause and effect reasoning using ATOMIC

- Can models generalize out of ATOMIC to previously unseen events ?
- Can models exploit the structure of nine inference dimensions ?

Sample from ATOMIC training data

3.7.2 Event2Mind: Commonsense Inference on Events, Intents, and Reactions

This section is based on [Rashkin et al.2018]

This work investigates a new commonsense inference task:

Example annotations of intent and reactions for 6 event phrases. Each annotator could fill in upto three free-responses for each mental state

Overview of Model Architecture

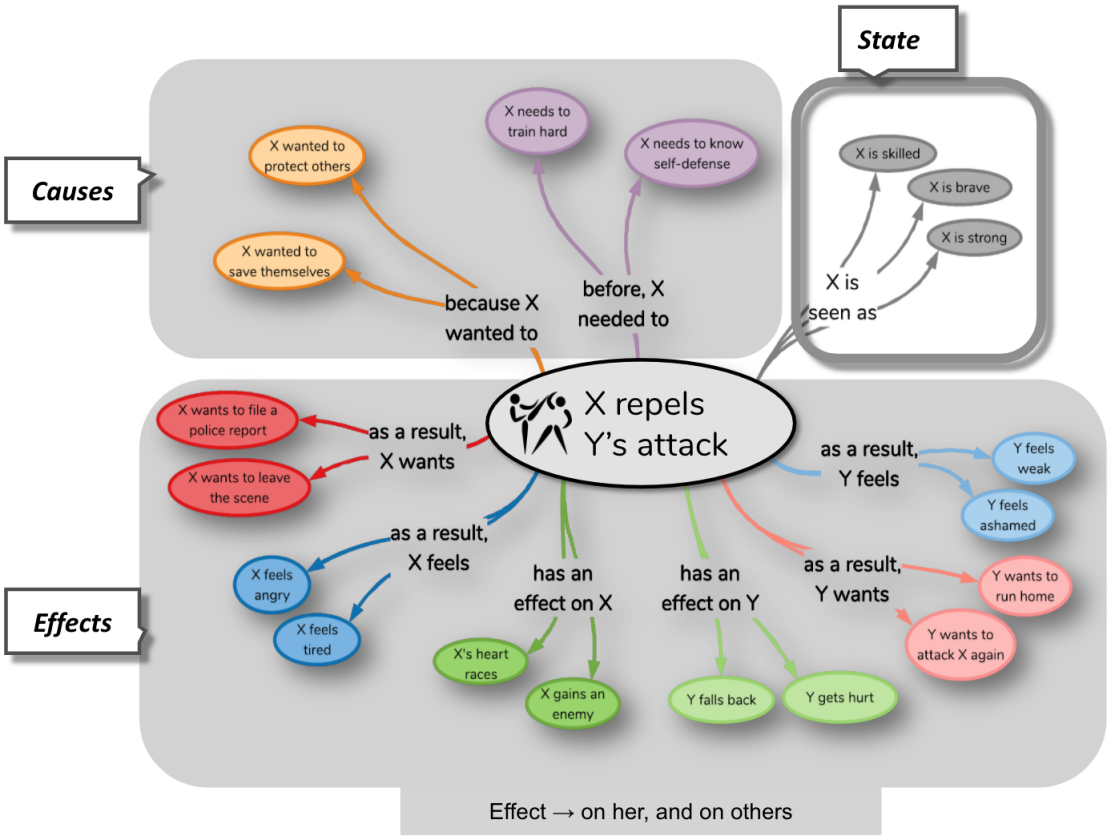

Figure 3: Nine inference dimension can be grouped into causes, effects and state related to the event [4]

Event: PersonX cries a lot

Annotated causes for PersonX

Because PersonX wanted (`xIntent`): *Annotators agreed that PersonX doesn't willingly cause this event.*
Before, PersonX needed (`xNeed`): did a mistake; goes through a big loss

Annotated attributes of PersonX

PersonX is seen as (`xAttr`): emotional; maudlin; childish; scared

Annotated effects on PersonX

As a result, PersonX feels (`xReact`): sad; depressed; emotional; sad; depressed; upset
As a result, PersonX wants (`xWant`): to gain; hard work; to cheer up; to remain content
Has an effect on PersonX (`xEffect`): get comforted by others; become grateful to her friends

Annotated effects on others

As a result, others feel (`oReact`): *Annotators agreed that this event doesn't emotionally affect others.*
As a result, others want (`oWant`): *Annotators agreed that this event doesn't make others want to do anything.*
Has an effect on others (`oEffect`): become concerned; become good friends with PersonX

Figure 4: Sample from ATOMIC training data [5]

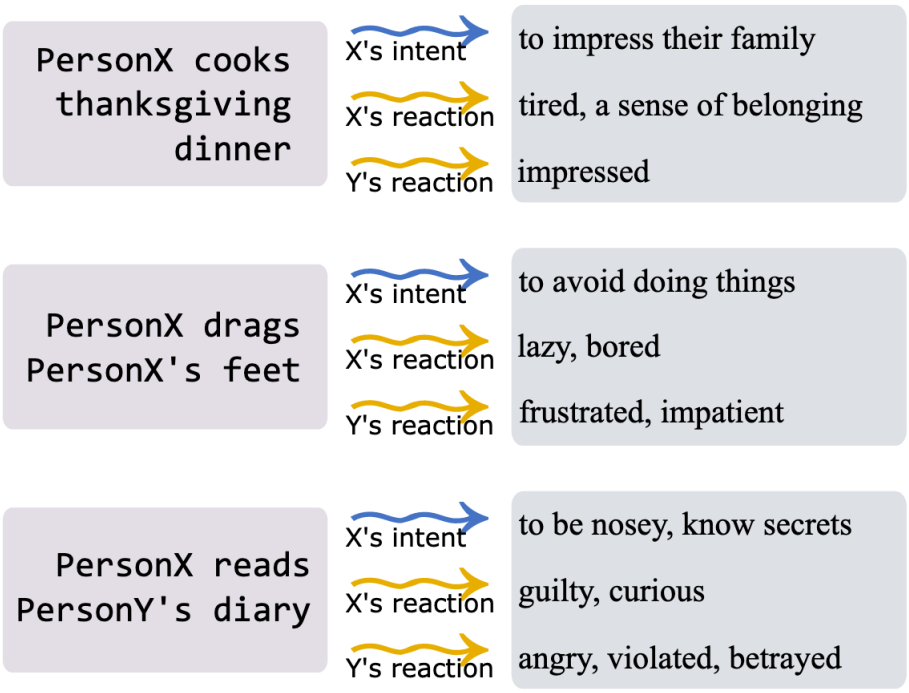

Figure 5: Examples of commonsense inference on mental states of event participant [6]

PersonX's Intent	Event Phrase	PersonX's Reaction	Others' Reactions
to express anger to vent their frustration to get PersonY's full attention	**PersonX starts to yell at PersonY**	mad frustrated annoyed	shocked humiliated mad at PersonX
to communicate something without being rude to let the other person think for themselves to be subtle	**PersonX drops a hint**	sly secretive frustrated	oblivious surprised grateful
to catch the criminal to be civilized justice	**PersonX reports ___ to the police**	anxious worried nervous	sad angry regret
to wake up to feel more energized	**PersonX drinks a cup of coffee**	alert awake refreshed	NONE
to be feared to be taken seriously to exact revenge	**PersonX carries out PersonX's threat**	angry dangerous satisfied	sad afraid angry
NONE	**It starts snowing**	NONE	calm peaceful cold

Figure 6: Example annotations of intent and reactions for 6 event phrases [7]

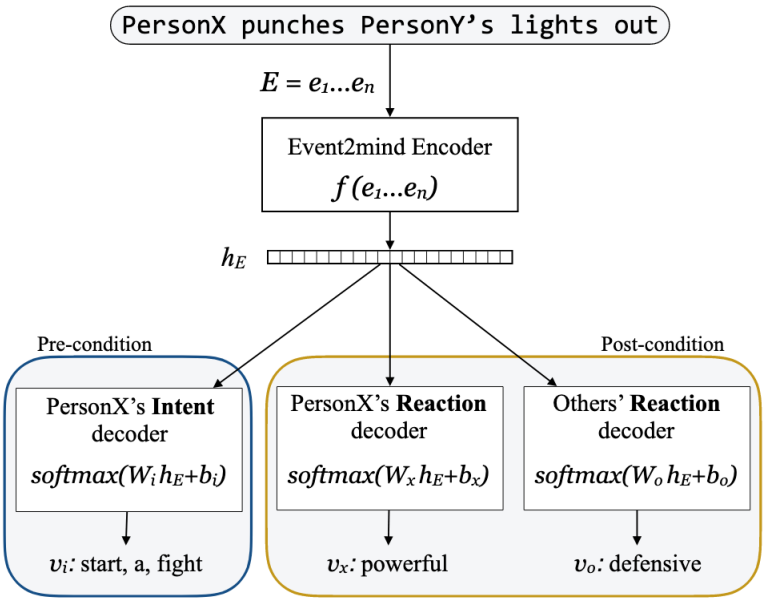

Figure 7: Overview of Model Architecture [8]

References

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, pages 9505–9515.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1885–1894. JMLR. org.

Hannah Rashkin, Maarten Sap, Emily Allaway, Noah A Smith, and Yejin Choi. 2018. Event2mind: Commonsense inference on events, intents, and reactions. *arXiv preprint arXiv:1805.06939*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " 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*, pages 1135–1144.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. 2014. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3319–3328. JMLR. org.

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity checks for saliency maps. In Advances in Neural Information Processing Systems. pages 9505–9515.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of in- terpretable machine learning. arXiv preprint arXiv:1702.08608 .

Tameru Hailesilassie. 2016. Rule extraction algorithm for deep neural net- works: A review. arXiv preprint arXiv:1610.05267 .

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep resid- ual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. pages 770–778.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. arXiv preprint arXiv:1902.10186 .

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pages 1885–1894.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2015. Visualizing and understanding neural models in nlp. arXiv preprint arXiv:1506.01066 .

Zachary C Lipton. 2016. The mythos of model interpretability. arXiv preprint arXiv:1606.03490 . 66Gary Mataev, Michael Elad, and Peyman Milanfar. 2019.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. 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. ACM, pages 1135–1144.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pages 856–865.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: Training differentiable models by constraining their explanations. arXiv preprint arXiv:1703.03717 .

Gregor PJ Schmitz, Chris Aldrich, and Francois S Gouws. 1999. Ann-dt: an algorithm for extraction of decision trees from artificial neural networks. IEEE Transactions on Neural Networks 10(6):1392–1401.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. 2017. Grad-cam: Visual expla- nations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision. pages 618–626.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carls- son. 2014. Cnn features off-the-shelf: an astounding baseline for recogni- tion. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pages 806–813.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In Proceed- ings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pages 3145–3153. 67

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep in- side convolutional

networks: Visualising image classification models and saliency maps. arXiv preprint arXiv:1312.6034 .

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viegas, and Martin Wattenberg. 2017. Smooth-grad: removing noise by adding noise. arXiv preprint arXiv:1706.03825 .

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribu- tion for deep networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, pages 3319–3328.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding con- volutional networks. In European conference on computer vision. Springer, pages 818–833.

Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Tor- ralba. 2016. Learning deep features for discriminative localization. In Proceedings of the IEEE conference on computer vision and pattern recog- nition. pages 2921–2929.