Natural Language Processing (2025), **31**, pp. 842–873
doi:10.1017/nlp.2024.41

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Thought flow nets: From single predictions to trains of model thought

Hendrik Schuff[1,3], Heike Adel[2] and Ngoc Thang Vu[3]

[1]Ubiquitous Knowledge Processing Lab, Technical University of Darmstadt, Darmstadt, Germany, [2]Hochschule der Medien, Stuttgart, Germany, and [3]Institut für Maschinelle Sprachverarbeitung, University of Stuttgart, Stuttgart, Germany
Corresponding author: Hendrik Schuff; Email: hendrik.schuff@tu-darmstadt.de

(Received 2 May 2023; revised 19 April 2024; accepted 16 May 2024; first published online 6 September 2024)

Abstract

When humans solve complex problems, they typically construct, reflect, and revise sequences of ideas, hypotheses, and beliefs until a final decision or conclusion is reached. Contrary to this, current machine learning models are mostly trained to map an input to one single and fixed output. In this paper, we investigate how we can equip models with the ability to represent, construct, and evaluate a second, third, and *k*-th thought within their prediction process. Drawing inspiration from Hegel's dialectics, we propose and evaluate the *thought flow* concept which constructs a sequence of predictions. We present a *self-correction mechanism* which (a) is trained to estimate the model's correctness and which (b) performs iterative prediction updates based on the gradient of the correctness prediction. We introduce our method focusing initially on question answering (QA) and carry out extensive experiments which demonstrate that (i) our method is able to correct its own predictions and that (ii) it can improve model performance by a large margin. In addition, we conduct a qualitative analysis of thought flow correction patterns and explore how thought flow predictions affect users' human-AI collaboration in a crowdsourcing study. We find that (iii) thought flows improve user performance and are perceived as more natural, correct, and intelligent regarding single and/or top-3 predictions.

Keywords: Machine learning; question answering; philosophy

1. Introduction

A majority of currently popular classification models map a specific input $\mathbf{x}$ (e.g., a token or a sentence) to an output $\hat{\mathbf{y}}$ (Bishop 2006) where $\hat{\mathbf{y}}$ can be a class, a sequence (e.g., a generated text) or an answer span extracted from a text context, for example. Typically, the $\mathbf{x} \rightarrow \hat{\mathbf{y}}$ mapping involves various modulations and abstractions of $\mathbf{x}$ in a latent space (e.g., hidden layers of a neural network) but does not support variations or trajectories of $\hat{\mathbf{y}}$.[a] Humans, on the other hand, rarely come to a single decision right away but follow a complex thought process that involves reflecting on initial decisions (taking into consideration various constraints, such as knowledge, beliefs, and intuition), comparing different hypotheses, or resolving contradictions.

While the human "trains-of-thought" process has been studied extensively in cognitive sciences and philosophy—one particular example being Hegel's dialectics (Maybee 2020)—"trains-of-thought" theories have not been further explored by the machine learning community. However,

Article updated 16 June 2025.
[a]Trajectories in the sense of a path through the space of probability distributions and, thereby, model predictions.

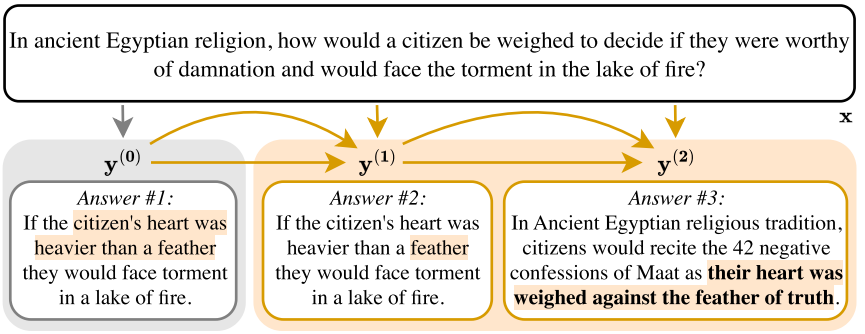


Figure 1. In contrast to the vanilla approach of mapping an input to an output in a single step (grey box), we propose a method that allows models to sequentially "reconsider" and update their predictions through a "*thought flow*" extension which can correct an incorrect (false) answer. In this (real) QA example, the orange box marks our thought flow extension, which corrects a flawed answer in two steps and ultimately returns the correct ground-truth answer (marked in **bold**).

with increasingly complex tasks that have large output spaces (such as question answering (QA)[b]), or tasks that require multiple reasoning steps (such as multi-hop QA), pose nontrivial practical challenges because learning to directly hit the right prediction in one shot might be more difficult than to learn to self-correct an initial prediction iteratively.

In this paper, we propose and evaluate a "*thought flow*" method for iterative self-correction of predictions via sequences of interdependent probability distributions with the goal of (a) increasing prediction performance and (b) providing a perspective onto the model's "reasoning" path that can provide explanatory value to humans. Furthermore, we propose a simple *correction module* to implement this concept. It can be used on top of any model that provides output logits of one or multiple distributions. In particular, it is inspired by the three moments of Hegel's dialectics which we map onto forward and backward passes in a model architecture, trained to judge whether the predicted class distribution corresponds to a correct prediction.

In our experiments on QA, we demonstrate our method's ability to self-correct incorrect (false) answer span predictions and identify qualitative patterns of self-correction, such as answer span reductions or extensions. Fig. 1 shows a real example of a thought flow which corrects a prediction ($\mathbf{y^{(0)}}$) output by a standard model to a new prediction ($\mathbf{y^{(2)}}$) via two steps, namely an answer span reduction and a cross-sentence answer jump. We find that our method can improve performance by up to 9.6% (absolute) in F_1 on a QA data set.

Finally, we assess the impact of thought flow predictions on human users within a crowdsourcing study. We find that thought flow predictions are perceived as significantly more correct, understandable, helpful, natural, and intelligent than single-answer predictions and/or top-3 predictions and they result in the overall best user performance without increasing completion times or mental effort.

In summary, our main contributions consist of (i) a formalization of a "thought flow" concept inspired by Hegel's dialectics, (ii) a novel correction module and a corresponding gradient-based update scheme to incorporate a thought flow into state-of-the-art Transformer networks, (iii) experiments on QA that demonstrate its strong correction capabilities and identify qualitative patterns of self-correction, (iv) a crowdsourcing user study that demonstrates that thought flows can improve perceived system performance as well as actual real-world user performance using the system, and (v) a demonstration of how our thought flow method can be applied beyond natural language processing using the example of a vision task.

[b]For example, the Longformer QA model (Beltagy et al. 2020) in the base-4096 variant (available at https://huggingface.co/allenai/longformer-base-4096) can output 16 M possible answer spans.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

2. Thought flow networks

In this section, we present background on Hegel's dialectics (Section 2.1), formalize thought flows based on it (Section 2.2), and describe a concrete implementation for QA (Section 2.3).

2.1 Inspiration: Hegel's dialectics

Drawing inspiration from Hegel's dialectics, our proposed method enables an existing model architecture to reflect and refine its predictions. In the following, we introduce the fundamental notion of the three "moments" in Hegel's dialectics and describe how we designed our thought flow concept. We provide further background on Hegel's dialectics in Appendix A.

Hegel's dialectics distinguishes three moments: (i) the *moment of understanding*, (ii) the *dialectical moment*, and (iii) the *speculative moment*. The moment of understanding refers to the initial, "seemingly stable" determination of a concept, object, or idea. In the second moment, the ostensible stability is lost due to the one-sidedness or restrictedness of the initial determination causing it to *sublate*[c] itself into its own negation. The speculative moment ultimately unifies and resolves the contradictory determinations by negating the contradiction (Maybee 2020).[d]

2.2 Formalization of thought flow concept

We now translate the high-level description of these three moments into a simplified mathematical setting that can be implemented in any (neural) model that uses a vector-valued representation of the input (such as an embedding) and outputs (tuples of) logits. In particular, we formalize and embed Hegel's dialectics in a framework with which we can obtain an initial "thought" vector and update it iteratively via the three "moments." Table 1 provides an overview of these moments and their corresponding elements in our thought flow method. In the following, we discuss these in detail.

Note that our formalization of Hegel's dialectics is approximative in nature, intended to inspire the development of novel machine learning models in general and thought flow nets in particular. For a detailed discussion of a formalization of Hegel's Dialectics and its logical status, we refer to, i.a., Ficara and Priest (2023), Nuzzo (2023), and Priest (2023).

Thought. We represent a *thought* as $\hat{\mathbf{z}} \in Z$, a vector of logits corresponding to a model's prediction and $Z \subseteq \mathbb{R}^c$ being the logit space of a prediction.[e] $\hat{\mathbf{z}}$ serves as a representation of the model's "decision state" as it captures implicit information about the most probable output (i.e., the decision itself) but also possible alternatives and uncertainty estimates encoded in the respective probability distribution. We want to emphasize that this representation of a "thought" should be understood as a metaphor that is necessarily lossy and simplistic and not as a neurologically plausible description.

Moment of Understanding. The first moment relates to an initial, seemingly stable determination of a concept, object, or idea in the model. We capture the moment of understanding through the initial value of $\hat{\mathbf{z}}^{(\mathbf{0})}$, which is the output of a prediction function $f_{\text{pred}} : \Phi \to Z$ applied to a model with an encoded input $\phi(\mathbf{x})$, an encoding function $\phi : \mathbb{R} \to \Phi$, and an encoding space $\Phi \subseteq \mathbb{R}^e$ (see Fig. 2(a)) where e denotes the dimensionality of the encoding space. Concretely, our formalization of the moment of understanding thus is the first model prediction represented as a probability distribution over the class labels.[f]

[c] Or "transcends" in the sense of "aufheben" (Ficara and Priest 2023).

[d] We discuss the common relation of the three moments and the thesis-antithesis-synthesis triad in Appendix A.1

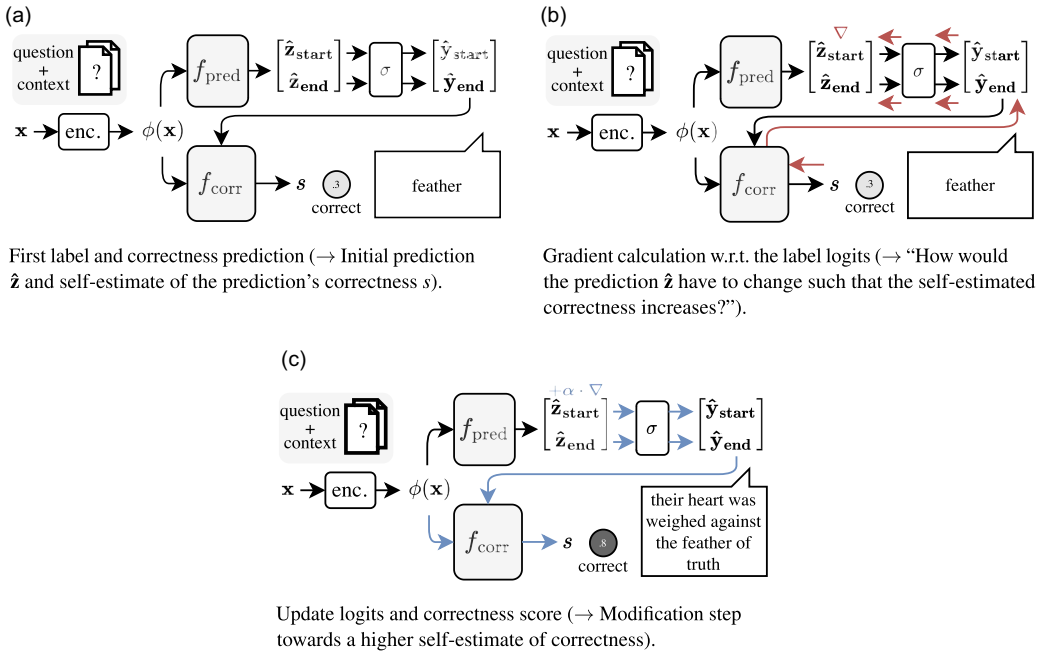
[e] We choose $\hat{\mathbf{z}}$ over $\hat{\mathbf{y}}$ because we can modify logits in an energy space without normalization in probability space.

[f] Note that we use the superscript to indicate the prediction's position within the thought flow and $\hat{\mathbf{z}}^{(\mathbf{0})}$ denotes the initial, unmodified model prediction.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table 1. Overview of the concepts from Hegel's dialectics which we draw inspiration from (left), their main characteristics (middle), and their corresponding elements in our proposed thought flow method (right)

Dialectical concept	Description	Correspondence in thought flow
Moment of Understanding	initial, seemingly stable determination	normal forward pass class prediction using f_{pred} (see Fig. 2(a))
Dialectical Moment	stability lost due to one-sidedness of the determination	self-estimate of prediction correctness using f_{corr} and respective gradient into the direction of estimated prediction improvement (see Fig. 2(b))
Speculative Moment	unification of the initial determination and its negation from the dialectical moment	update of the prediction using a gradient step into the direction of self-estimated prediction improvement (see Fig. 2(c))



First label and correctness prediction (→ Initial prediction $\hat{\mathbf{z}}$ and self-estimate of the prediction's correctness s).

Gradient calculation w.r.t. the label logits (→ "How would the prediction $\hat{\mathbf{z}}$ have to change such that the self-estimated correctness increases?").

Update logits and correctness score (→ Modification step towards a higher self-estimate of correctness).

Figure 2. The steps of our prediction update scheme. The example shows the second answer change from Fig. 1. $\mathbf{x}$ refers to the model input and represents the question and its given textual context, "enc." denotes an encoder function (e.g., a Transformer model), f_{pred} maps the encoding $\phi(\mathbf{x})$ to logits that correspond to probability distributions over start and end positions of the respectively predicted answer. In addition to this standard model architecture, we propose the addition of a function f_{corr} that is trained to predict an estimate of a correctness score s (e.g., F_1 score) given $\phi(\mathbf{x})$ and the probability distributions predicted by f_{pred}.

Dialectical Moment. At the second moment, the determination of the concept, object, or idea becomes unstable due to the initial determination's one-sidedness or restrictedness. To model this, we apply a new function $f_{\text{corr}} : Z \times \Phi \to \mathbb{R}$ that differentiably maps $\hat{\mathbf{z}}^{(\mathbf{0})}$ to a correctness score $s \in \mathbb{R}$ which is an estimate of the quality of the model prediction corresponding to $\hat{\mathbf{z}}^{(\mathbf{0})}$ while being conditioned on $\phi(\mathbf{x})$. Intuitively, $f_{\text{corr}}(\hat{\mathbf{z}}^{(\mathbf{0})}, \phi(\mathbf{x}))$ quantifies an estimate of how good the current prediction corresponding to $\hat{\mathbf{z}}^{(\mathbf{0})}$ is given the model input corresponding to $\phi(\mathbf{x})$. Note that f_{corr} does not have access to the (unknown) ground-truth label, but only relies on the predicted logits and the input representation. Intuitively, f_{corr} thus quantifies how plausible the combination of

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

the prediction $\hat{\mathbf{z}}^{(\mathbf{0})}$ and the encoded input $\phi(\mathbf{x})$ is. This quantification can rely on (a) individual features of the prediction (e.g., a prediction with a very high entropy can be more likely to be erroneous), (b) individual features of the encoded input (e.g., instances from a specific subdomain can be harder to predict than instances from another subdomain), or (c) the combination of both (e.g., a high entropy in a certain subdomain can signal an error).

Using our definition of f_{corr}, we formalize the dialectical moment with the gradient of the correctness score with respect to $\hat{\mathbf{z}}^{(\mathbf{0})}$, that is $\nabla^T_{\hat{\mathbf{z}}^{(0)}} s$ (see Fig. 2(b)) where "$\cdot^T$" denotes transposition and s the correctness score. The gradient calculation determines how the thought $\hat{\mathbf{z}}^{(\mathbf{0})}$ needs to change to receive a higher correctness self-estimate. The gradient, therefore, represents the determination's instability: as it creates tension away from the current $\hat{\mathbf{z}}^{(\mathbf{i})}$ towards the new $\hat{\mathbf{z}}^{(\mathbf{i+1})}$, it is destabilized, thus "negating" the initial determination in the sense that it modulates and thereby invalidates its initial value.

Speculative Moment. The third moment unites the initial determination with the negation[g] from the dialectical moment. We formalize this by modifying $\hat{\mathbf{z}}^{(\mathbf{0})}$ with a gradient step into the direction of an increased estimate of the correctness score s that yields

$$\hat{\mathbf{z}}^{(\mathbf{1})} := \hat{\mathbf{z}}^{(\mathbf{0})} + \alpha^{(0)} \cdot \nabla^T_{\hat{\mathbf{z}}^{(0)}} s \tag{1}$$

where $\alpha^{(0)}$ is a (potentially dynamic) step width and $\hat{\mathbf{z}}^{(\mathbf{1})}$ again constitutes the subsequent first moment of the next iteration (see Fig. 2(c)).

Iteration. Iterative application of the dialectical and the speculative moment's formalizations yield a sequence of logits $\left(\hat{\mathbf{z}}^{(\mathbf{k})}\right)_{k=0}^{N}$ and predictions $\left(\hat{\mathbf{y}}^{(\mathbf{k})}\right)_{k=0}^{N}$ where k iterates between the initial prediction for $k = 0$ and the ultimate prediction for $k = N$ by applying N gradient updates resulting from Equation (1).

In the following, we present a concrete implementation of this abstract formalization, focusing on the QA domain.

2.3 Implementation on Transformers for QA

Fig. 2 visualizes our formalization around the QA example introduced in Fig. 1. We now discuss QA-related implementation details.

2.3.1 Choosing parameters and functions

To apply our abstract thought flow method to a concrete model architecture, we have to (a) determine how we structure the model prediction logit vector $\hat{\mathbf{z}}$; (b) choose an input representation $\phi(\mathbf{x})$ (that is passed to f_{pred} as well as f_{corr}); (c) choose a parametrization of the self-estimated correctness score prediction function f_{corr}; and (d) define what the score s measures. In the following, we describe how these aspects can be realized in a Transformer-based QA model.

Composing $\hat{\mathbf{z}}$. In extractive QA, a typical approach to answer span extraction from a context of L tokens is to use two probability distributions: (i) $\hat{\mathbf{y}}_{\text{start}} \in [0, 1]^L$ that assigns to each token a probability of starting the answer span, and (ii) a respective end token distribution $\hat{\mathbf{y}}_{\text{end}} \in [0, 1]^L$ for the probability of ending the answer span. To match our previously defined formalization, we define

[g]Note that *negation* refers to its meaning in Hegel's dialectic here and does not directly relate to an arithmetic or logical meaning of negation.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

$\hat{\mathbf{z}}^{(\mathbf{i})} := \left[\hat{\mathbf{z}}^{(\mathbf{i})}_{\text{start}}\ \hat{\mathbf{z}}^{(\mathbf{i})}_{\text{end}}\right]^{\mathrm{T}}$ which is linked to the corresponding probability distributions via the softmax function σ:

$$\begin{aligned}\hat{\mathbf{y}}^{(\mathbf{i})} &:= \left[\hat{\mathbf{y}}^{(\mathbf{i})}_{\text{start}}\ \hat{\mathbf{y}}^{(\mathbf{i})}_{\text{end}}\right]^{\mathrm{T}} \\ &= \left[\sigma\left(\hat{\mathbf{z}}^{(\mathbf{i})}_{\text{start}}\right)\ \sigma\left(\hat{\mathbf{z}}^{(\mathbf{i})}_{\text{end}}\right)\right]^{\mathrm{T}}.\end{aligned}$$

Input Representation $\phi(\mathbf{x})$. In contrast to Transformer-based classification models that conventionally rely on the embedding of the [CLS] token, typical Transformer-based QA models apply a linear function on top of each token's embedding that maps the embedding to a start and an end logit. We follow this convention and define

$$\phi(\mathbf{x}) := [\mathbf{e_1}, \mathbf{e_2}, \ldots, \mathbf{e_L}] \in \mathbb{R}^{d \times L} \tag{2}$$

that is, as the sequence of L contextualized embeddings with embedding dimension d.

Choosing f_{corr}. To represent the input within f_{corr}, we need a representation of $\phi(\mathbf{x})$ that focuses on the relevant parts of the (potentially very long) input that were relevant to predict the start and end logits. We thus choose a weighted average over all token embeddings to retain as much as possible of the important information from the input while heavily reducing its available representation dimensionality to a single vector. As weights, we choose the element-wise product of the predicted start and end probabilities. We thus define a modified input encoding $\tilde{\phi}^{(i)}(\mathbf{x}) \in \mathbb{R}^d$ where d denotes the dimension of the embeddings[h] as follows:

$$\tilde{\mathbf{w}}^{(\mathbf{i})} := \left(\hat{\mathbf{y}}^{(\mathbf{i})}_{\text{start}} \odot \hat{\mathbf{y}}^{(\mathbf{i})}_{\text{end}} + \varepsilon \cdot \mathbf{1}\right) \in \mathbb{R}^L \tag{3}$$

$$\tilde{\phi}(\mathbf{x})^{(i)} := \phi(\mathbf{x}) \cdot \frac{\tilde{\mathbf{w}}^{(\mathbf{i})}}{\Sigma_j \tilde{\mathbf{w}}^{(\mathbf{i})}_j} \in \mathbb{R}^d \tag{4}$$

where ε is a small constant that ensures that we do not divide by zero, e_i is the embedding of the i-th token, $\odot$ is element-wise multiplication, and L is the maximum number of tokens in the context. This modified input representation $\tilde{\phi}(\mathbf{x})^{(i)}$ can be regarded to be a dynamic perspective onto $\phi(\mathbf{x})$ that highlights these parts of $\phi(\mathbf{x})$ that are most important to the model's answer prediction. The intuition behind this is that the correction module should have access to all information about the context that the prediction model focused on. Based on initial empirical findings, we choose to use a two-layer multi-layer perceptron (MLP) with scaled exponential linear unit (SELU) activation (Klambauer *et al.* 2017) to map the concatenated vector

$$\left[\text{dropout}\left(\tilde{\phi}^{(i)}\right)\ \hat{\mathbf{z}}^{(\mathbf{i})}_{\text{start}}\ \hat{\mathbf{z}}^{(\mathbf{i})}_{\text{end}}\right]^{\mathrm{T}} \in \mathbb{R}^{d+2\cdot L} \tag{5}$$

to an estimated correctness score s. Note that f_{corr} does not receive the decoded answer text but uses the start and end logits directly to provide differentiability.

Correctness Score s. Following standard evaluation metrics for QA, we use the F_1-score of the predicted answer as the correctness score target that f_{corr} is trained to predict.

2.3.2 Training

To train f_{corr}, we freeze the parameters of f_{pred}. Then, we pass the training instances through the whole model (including ϕ, f_{pred}, and f_{corr}) as shown in Fig. 2(a) to obtain the target of the predicted correctness score s. We determine the ground-truth correctness score by calculating the F_1-score between the ground-truth answer and the answer prediction from f_{pred}. We define the correctness

[h] For example, 768 for BERT-base (Devlin *et al.* 2019).

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

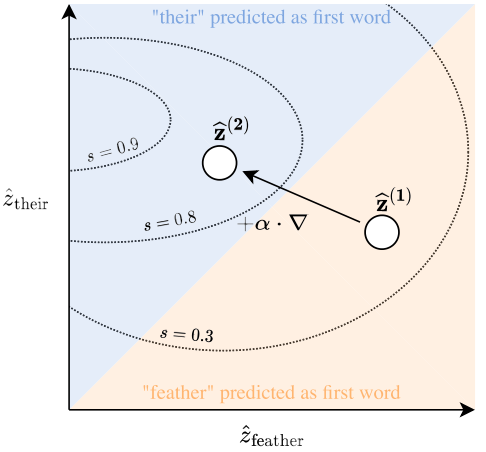


Figure 3. Simplified visualization of the modification step (during inference) shown in Fig. 2 depicted within a cut through the logits space. The axes correspond to the elements of $\hat{\mathbf{z}}_{\text{start}}$ that represent the index positions of the words "feather" and "their" as shown in the example in Fig. 1. The dotted isolines correspond to the self-estimated correctness scores obtained from f_{corr}. Before the modification, the system's prediction corresponding to $\hat{\mathbf{z}}^{(1)}$ would be an answer starting at "feather." The gradient ∇ points in the direction of an improvement of self-estimated correctness of $\hat{\mathbf{z}}^{(1)}$. After a gradient step into the direction of ∇, the change in logits towards $\hat{\mathbf{z}}^{(2)}$ leads to a shift of the answer start position. The modification behavior emerges solely from the logit modifications and can lead to complex modification patterns (see Section 3.3).

estimate prediction loss as the mean squared error between the calculated score, and the predicted s and train f_{corr} to minimize it. Overall, we thus train f_{corr} to score how correct a model prediction (represented by the start and end logits) is given a model input (represented by the condensed input encoding $\tilde{\phi}(\mathbf{x})$) and use the model's predictions on the training set to generate ground-truth correctness scores (using F_1-score).

2.3.3 Inference

At inference time, we encode a new input and predict (i) the answer start and end logits using f_{pred} and (ii) an estimated F_1-score s of the predicted answer span using the correction module f_{corr} as shown in Fig. 2(a). Instead of directly using the initial logits as the model's prediction—as would be done in a standard model—we iteratively update the logits with respect to the estimated correctness score's gradient following our formalization from Section 2.2 as shown in Fig. 2(b),(c). Fig. 3 depicts a simplified visualization of one gradient update that changes the model's prediction.

Update Rule. As described in Section 2.2, we aim at modifying $\hat{\mathbf{z}}^{(i)}$ so that the correction module assigns an increased correctness (i.e., F_1-score in our QA application). To apply Equation (1), we have to define how the step size α is chosen in our QA application. We choose α such that a predefined probability mass δ is expected to move. To this end, we first take a probing step of length one, calculate the distance as the L_1 norm between the initial distribution and the probe distribution, and choose the step width $\alpha \in \mathbb{R}^+$ such that it scales the linearized L_1 distance to the hyperparameter δ as follows:

$$\alpha := \left[\frac{\delta}{\left\| \sigma\left(\hat{\mathbf{z}}^{(i)}\right) - \sigma\left(\hat{\mathbf{z}}^{(i)} + \nabla^T_{\hat{\mathbf{z}}^{(i)}} s\right) \right\|_1 + \varepsilon} \right] \tag{6}$$

where $\sigma(\cdot)$ denotes the softmax function, s is the correctness score as defined above, and $\varepsilon \in \mathbb{R}^+$ is a small constant for numerical stability.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Monte Carlo Dropout Stabilization. The gradient $\nabla_{\hat{\mathbf{z}}^{(i)}} s$ is deterministic but it can, as we found in our preliminary experiments, be sensitive to small changes in the input representation $\phi(\mathbf{x})$. We therefore stabilize our correction gradient estimation by *sampling* and averaging gradients instead. For this, we use the dropped-out input encoding from Equation (5) and sample five gradients for every step using MCDrop (Gal and Ghahramani 2016).

3. Question answering experiments

3.1 Data, model, and training

3.1.1 Data set

We choose the HOTPOTQA data set (distractor setting) (Yang *et al.* 2018) to evaluate our models because it contains complex questions that require multi-hop reasoning over two Wikipedia articles. In the distractor setting, the model is "distracted" by eight irrelevant articles which are passed to the model alongside a pair of relevant articles. In addition to yes/no/answer span annotations, HOTPOTQA also provides explanation annotations in the form of binary relevance labels over the paragraphs of the relevant articles which we do not use when training our models. As the public test set is undisclosed, we use the official validation set as our test set and a custom validation set with 10k instances sampled from the training set leaving 80,564 instances for training.

3.1.2 Base model

Our underlying QA model is Longformer-large[i] (Beltagy, Peters, and Cohan 2020) with a final linear layer that maps token embeddings to start and end logits. The model reaches 63.5% F_1 (SD = 0.6) on the HOTPOTQA validation set averaged over three random seeds and can handle input lengths of up to 4096 tokens which enables us to feed in the entire context as a single instance without truncation. The model's input is a single token sequence that contains the question followed by the answer context (i.e., the concatenation of 8 + 2 Wikipedia articles). The model outputs two distributions over the input tokens (i.e., two 4096-dimensional distributions), namely (a) one for the answer start position and (b) another for the answer end position, respectively. In this commonly used extractive QA setting, the model can choose its answer from any text span within the context. We prepend a "yes" and a "no" token to the context (instead of adding a categorical prediction head for these answer options) because doing so makes it easier to align distributions across answer options and text span options. In total, this model has 435 M parameters of which only 331k parameters are added by our MLP implementation of f_{corr}.

3.1.3 Training details

We first train the base models for five epochs on a single V100 GPU using a learning rate of 10^{-5}, an effective batch size of 64, an AdamW optimizer (Loshchilov and Hutter 2019), early stopping, and a cross-entropy loss on the start/end logits. We subsequently train the correction modules using the same setting but with the mean squared error loss function for F_1-score prediction training. Training a single model each took approximately three days. In the following, we report all results as averages over three random seeds including standard deviations.

3.2 Performance improvements

3.2.1 Performance over steps

Fig. 4(a) shows how F_1-scores per gradient scaling target δ evolve over 100 steps. We observe that small δ values lead to small F_1 improvements. While $\delta = 0.1$ consistently improves F_1-scores, all other δ values eventually deteriorate F_1-scores. The higher the δ value, the faster the F_1 decrease.

[i] https://huggingface.co/allenai/longformer-large-4096

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

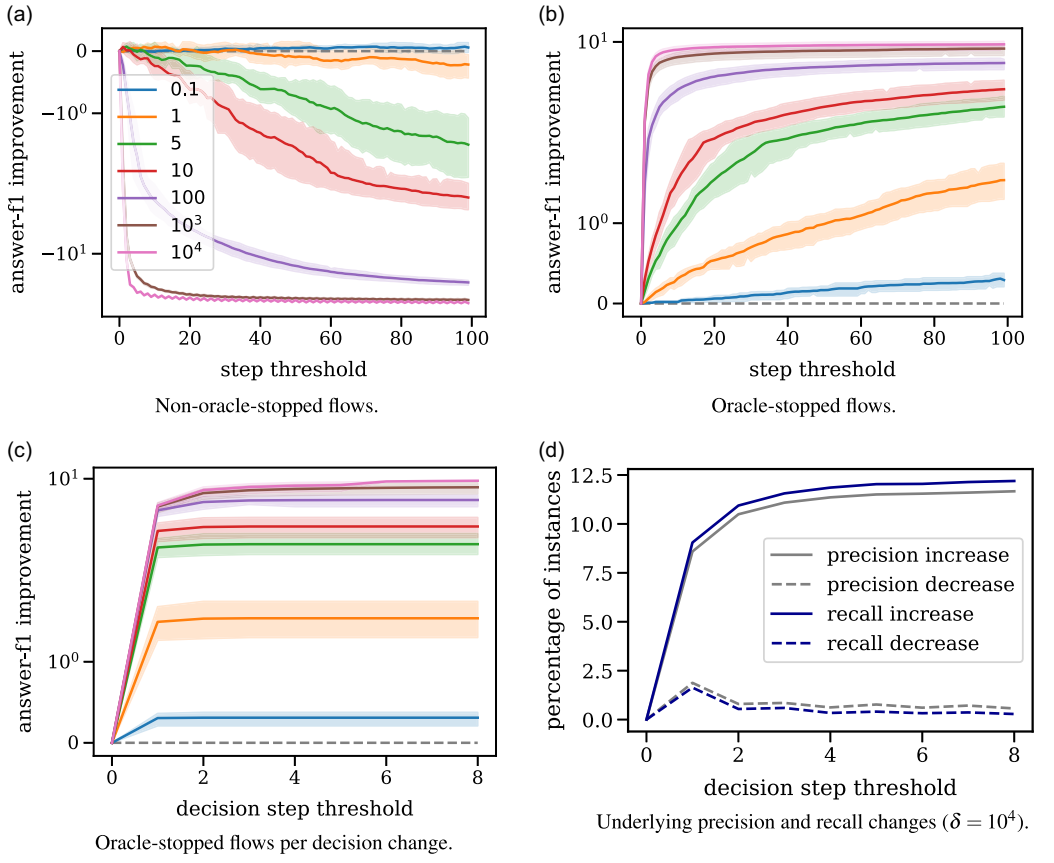


Figure 4. Thought flows with different gradient scaling targets δ averaged over three seeds of a QA model. Higher values for δ correspond to more aggressive decision changes. Without a stopping oracle that stops when the thought flow no longer improves an answer (top left), only $\delta = 0.1$ provides consistently stable, but very small F_1 improvements. With an oracle (top right), higher values for δ reach higher and faster F_1 improvements up to >9%. Nearly all performance gains are achieved by the first decision change (bottom left). A detailed analysis of flows using $\delta = 10^4$ shows that the observed F_1 improvements are the result of slight decreases and stronger increases in both precision and recall (bottom right). y axes of plots (a), (b), and (c) use a symlog scale. Improvements are reported as absolute F_1 score differences to the base model performance of 63.5% F_1.

We conclude that (i) very small δ values fail to offer notable performance gains and that (ii) a larger δ can improve performance at the beginning but then "overshoot" with their corrections. We hypothesize that a remedy to this tradeoff is to use larger δ values but stop the modification process at the right time.

3.2.2 Stopping via an oracle

To test the hypothesis that stopping modifications at the right time can unlock notable performance improvements, we introduce a stopping function which uses an oracle to stop the thought flow when the best F_1 performance level is detected. Fig. 4(b) shows that, with this oracle function, thought flows can reach performance improvements of up to 9.6% F_1 (SD = 0.61) corresponding to an absolute best average performance of 73.1% F_1 using $\delta = 10^4$.

Fig. 4(c) shows that almost all performance improvements are due to the first decision change within the thought flows. Moreover, answer spans improve constantly and do not randomly

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

shift across the context. This observation suggests that singular thought flow changes are highly effective and can make substantial corrections rapidly.

Fig. 4(d) shows that the observed F_1 improvements are the result of an overall increase in precision and recall. In turn, these increases are composed of small decreases in precision and recall that are outweighed by stronger increases across all decision step thresholds. Fig. 4(d) displays the fraction of instances for which precision/recall values decrease/increase for thought flows with $\delta = 10^4$. We provide the respective analyses for additional δ values in Figure B1 in Appendix B.2.

3.3 Thought flow patterns

In the following, we investigate which answer span modifications lead to the observed performance improvements. Note that we did not manually specify or limit the possible answer span modification types. Instead, all observed modification patterns emerge solely based on changes in the underlying start and end position distributions, and these changes are induced by gradient steps in the direction of improved self-estimated prediction correctness.

We randomly sample 150 instances from the subset of the official validation split for which the thought flow changed the initial answer prediction. We identify six (non-exclusive) basic modification patterns as well as two additional composed modification patterns that all emerge from the combination of correctness self-estimation and iterative prediction updates and show selected examples in Table 2. We provide additional examples for each correction pattern in Table B1. Further, Table 3 shows additional thought flow examples using three modification steps.

Cross-Sentence. With 52.7%, this is the most frequent type of modification pattern. The thought flow moves the predicted answer span from one sentence to another.

Span Reduction. The thought flow can shorten the predicted span to modify the answer.

Span Extension. Similarly, the thought flow can also expand a predicted answer span to modify it.

In-Sentence. Beyond in-sentence span reduction/extension, the thought flow can also shift between non-overlapping spans within a sentence.

Entity Refinement. In this modification pattern, the thought flow keeps predicting the same entity but switches the predicted answer to an alternative mention of the entity.

Logic Hops. The thought flow performs a step-wise reasoning that first resolves the first step of a two-step reasoning structure before jumping to the second step, that is the modified answer.

Combinations. We observe various combinations of the aforementioned patterns. A model can, for instance, jump between sentences, refine entities, and reduce the answer span.

Sequential Modifications. Modifications can also occur sequentially as shown in the examples in Table 3. While the example in the upper part of Table 3 demonstrates a combination of a cross-sentence modification followed by a span reduction modification, the example in the lower part illustrates how a span extension modification can iteratively improve a prediction. We additionally observe flow patterns with very high numbers of answer modifications. These typically correspond to sequences of modifications in which the answer periodically alternates between two or three answer alternatives or in which the change of answers exhibits seemingly chaotic behavior. Table 4 shows two example flows that contain 45 and 12 decision changes. The first example shows a frequently observed long-flow modification pattern in which the initial decision is followed by a

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table 2. Emergent thought flow modification patterns identified in 150 randomly sampled thought flows using $\delta = 1$. The **correct answer** is marked in bold, and the predicted answer per flow step is marked in orange

Modification pattern examples
cross-sentence (52.7%)
Question: Who is older Danny Green or James Worthy?
(1) Daniel Richard "Danny" Green, Jr. (born June 22, 1987) is an American professional basketball player for the San Antonio Spurs of the National Basketball Association
(2) **James Ager Worthy** (born February 27, 1961) is an American professional basketball coach and former player, commentator, television host, and analyst
span reduction (23.3%)
Question: What philosophy related to creationism is Paul Nelson noted for?
(1) Paul A. Nelson (born 1958) is an American philosopher of science noted for his advocacy of young earth creationism and **intelligent design**
(2) Paul A. Nelson (born 1958) is an American philosopher of science noted for his advocacy of young earth creationism and **intelligent design**
span extension (21.3%)
Question: Ronald Reagan and George H. W. Bush both held which position in office?
(1) The presidency of Ronald Reagan began on January 20, 1981, when Ronald Reagan was inaugurated as **President of the United States**, and ended on January 20, 1989
(2) The presidency of Ronald Reagan began on January 20, 1981, when Ronald Reagan was inaugurated as **President of the United States** , and ended on January 20, 1989
in-sentence (7.3%)
Question: When was the stadium that held the 2015 Magyar Kupa demolished?
(1) The stadium was closed in 2016 and demolished in **2017** to give place to the new Ferenc Puskas Stadium
(2) The stadium was closed in 2016 and demolished in **2017** to give place to the new Ferenc Puskas Stadium
entity refinement (4.5%)
Question: Which host of Sunday Night Safran has the hebrew first name Yehoshua?
(1) John Michael Safran (Hebrew: "Yehoshua Safran" ; born 13 August 1972) is an Australian radio personality, satirist, documentary maker, and author, known for combining humour with religious, political, and ethnic issues
(2) It was hosted by **John Safran** and Catholic priest, Bob Maguire
logic hops (4%)
Question: Is the Pakistan fast bowler who joined the Kent County Cricket Club in June 2011 a left-hand or right-hand batsmans?
(1) Wahab Riaz (Punjabi, Urdu: ; born 28 June 1985) is a Pakistani cricketer
(2) He is a left-arm fast bowler and a **right-hand** batsman
combined (9.3%)
Question: Who was born in 1922 and published a book in 1985 by Delacorte Press?
(1) Kurt Vonnegut Jr. (November 11, 1922; April 11, 2007) was an American writer
(2) Galapagos is the eleventh novel written by American author **Kurt Vonnegut**

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table 3. Multi-step modification examples ($\delta = 1$). The **correct answer** is marked in bold, the predicted answer per flow step is marked in orange

Examples
Question: How many times did the man who coached the 1986-87 UNLV Runnin' Rebels fail to win 20 games in a season?
(1) He spent the majority of his career coaching with the UNLV Runnin' Rebels, leading them four times to the Final Four of the NCAA Men's Division I Basketball Tournament, winning the national championship in 1990
(2) Overall, he won over 700 games in his career, and only **twice** failed to win 20 games in a season
(3) Overall, he won over 700 games in his career, and only **twice** failed to win 20 games in a season
Question: Why did the CEO of the football team based in Denver, Colorado step down in 2014?
(1) He served as the Broncos CEO from his purchase of the club in 1984 until July 2014, when he stepped down as Broncos' CEO **due to the onset and progression of Alzheimer's disease**
(2) He served [. . .], when he stepped down as Broncos' CEO **due to the onset and progression of Alzheimer's disease**
(3) He served [. . .], when he stepped down as Broncos' CEO **due to the onset and progression of Alzheimer's disease**

Table 4. Examples of long thought flows. The table shows two questions for which the resulting thought flow contains 45 prediction changes and 12 decision changes respectively. In contrast to previous tables, the answer contexts are omitted and only the predicted answer spans are displayed. Different shades of orange background reflect repeated answer spans and highlights, for example, a 2-cycle for the first example

Examples
Question: Which other Mexican Formula One race car driver has held the podium besides the Force India driver born in 1990?
Sergio Perez → Adrian Sutil → Sergio → **Pedro Rodriguez**
Question: How are Ceephax Acid Crew and Squarepusher's music similar?
psychedelic → sound → acid house → sound → acid house → sound → acid house → music → acid house → sound → acid house → music → **electronic musician**

2-cycle between two alternative decisions before the answer changes to the final answer that was not predicted before.

While the previous examples showed thought flow modifications that result in a successful modification of a wrong or incomplete answer to the correct answer, thought flow modifications can also deteriorate answer correctness. We provide comparisons of successful and unsuccessful modifications for two frequent modification patterns (i.e., cross-sentence and span extension modifications) in Table 5 and Table 6. Overall, we do not observe a systematic correspondence between successful/unsuccessful modification and specific modification pattern types or

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table 5. Examples of successful (incorrect → correct) and unsuccessful cross-sentence modifications. The **correct answer** is marked in bold, and the predicted answer per flow step is marked in orange

Cross-sentence modifications
successful
Question: What other country does the league that Taylor Eric Kemp played in represents the sport's highest level besides the United States?
(1) The Honkbal Overgangsklasse (Dutch for "Baseball Transition League") is the second highest level of professional baseball in the Netherlands
(2) Major League Soccer is a men's professional soccer league, sanctioned by U.S. Soccer, that represents the sport's highest level in both the United States and **Canada**
unsuccessful
Question: What items are used to play both Lapta and Rounders?
1) Pesapallo (] ; Swedish: "boboll,", both names literally meaning "nest ball," also referred to as "Finnish baseball") is a fast-moving bat-and-ball sport that is often referred to as the national sport of Finland and has some presence in other countries including Germany, Sweden, Switzerland, Australia, and Canada's northern Ontario (the latter two countries have significant Nordic populations)
(2) Rounders is a striking and fielding team game that involves hitting a small, hard, leather-cased ball with a rounded end wooden, plastic, or metal bat
Ground-truth answer: "bat and ball"

Table 6. Examples of successful (incorrect → correct) and unsuccessful span extension modifications. The **correct answer** is marked in bold, and the predicted answer per flow step is marked in orange

Span extension modifications
successful
Question: The Kellock-Taschereau Commission was appointed by a prime minister who served how long in office?
(1) A Liberal with **21 years and 154** **days** in office, he was the longest-serving prime minister in Canadian history
(2) A Liberal with **21 years and 154 days** in office, he was the longest-serving prime minister in Canadian history
unsuccessful
What do the Rampur Greyhound and Borzoi have in common?
(1) The Borzoi (, literally "fast"), also called the Russian wolfhound (Russian: Russkaia psovaia borzaia), is a breed of domestic dog ("Canis lupus familiaris")
(2) The Borzoi (, literally "fast"), also called the Russian wolfhound (Russian: Russkaia psovaia borzaia), is a breed of domestic dog ("Canis lupus familiaris")
Ground-truth answer: "member of the sighthound family"

question characteristics. We provide additional examples of the discussed modification patterns in Appendix B.2.

4. Human evaluation

While the previous section showed that thought flows can yield complex prediction modification patterns and can reach promising performance gains, we now investigate how thought flow predictions affect human users in an AI-assisted QA task.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

4.1 Experiment design

We choose a within-subject design in which each participant is exposed to three variations of a QA system.

4.1.1 Conditions

We aim to assess the effect of the thought flow concept on users and therefore present the outputs of the oracle-stopped thought flow in one condition (**TF**) and compare it to two baseline conditions. As baselines, we use top-1 predictions (**SINGLE**) (to compare against standard models) and top-3 predictions (**TOP-3**) (to compare to an alternative approach to show several predictions). For all conditions, we present the predicted answer(s) along with the sentence in which they appear in the context.

4.1.2 Dependent variables

We study the effect of the condition (SINGLE, TF, and TOP-3) on a set of dependent variables. We include variables on a per-question level (after each question) and on a per-system level (after all questions of one condition).

The per-question variables include the following: (i) **human answer correctness**, (ii) **perceived model correctness**, (iii) **perceived understanding**, (iv) **perceived helpfulness**, and (v) **completion time**. The per-system variables include: (vi) **usability** using the Usability Metric for User Experience (UMUX) questionnaire (Finstad 2010, 2013), (vii) **mental effort** using the Paas scale (Paas 1992), (viii) **anthropomorphism** using the respective subscale of the Godspeed questionnaire (Bartneck *et al.* 2009), (ix) **perceived intelligence** using the subscale from the same questionnaire, and (x) **average completion time**.[j] We provide a list of all questionnaires in the appendix.

4.1.3 Apparatus

We sample 100 instances from the HOTPOTQA validation instances in which a thought flow using $\delta = 1$ made at least one prediction change.[k] From these, we sample 30 instances per participant and randomly assign the instances to three bins of 10 questions (one bin per condition).[l] We balance the six possible condition orders across participants and include three attention checks per participant. Fig. 5 shows our user study interface for the TF condition in the example of the previously discussed thought flow instance. We provide screenshots of all conditions' interfaces in the appendix.

4.2 Quantitative results

We use MTurk[m] to recruit US-based crowdworkers with >90% approval rate and the MTurk Masters qualification to be consistent with previous human evaluations of explainability on the HOTPOTQA data set (i.a., Schuff *et al.* 2023a). We collect responses from 55 workers.[n]

[j] We rely on the existing UMUX, Paas, and Godspeed scales as these are frequently used and well-studied in human-computer interaction and human-robot interaction research. We drop the robotics-specific item regarding "moving rigidly/elegantly" from the Godspeed subscale as it is not applicable to QA.

[k] If there is no prediction change, TF is identical to SINGLE.

[l] We statistically account for random effects of single questions.

[m] https://www.mturk.com/

[n] We filter out two participants who did not pass the attention checks and replace them with two additional responses.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

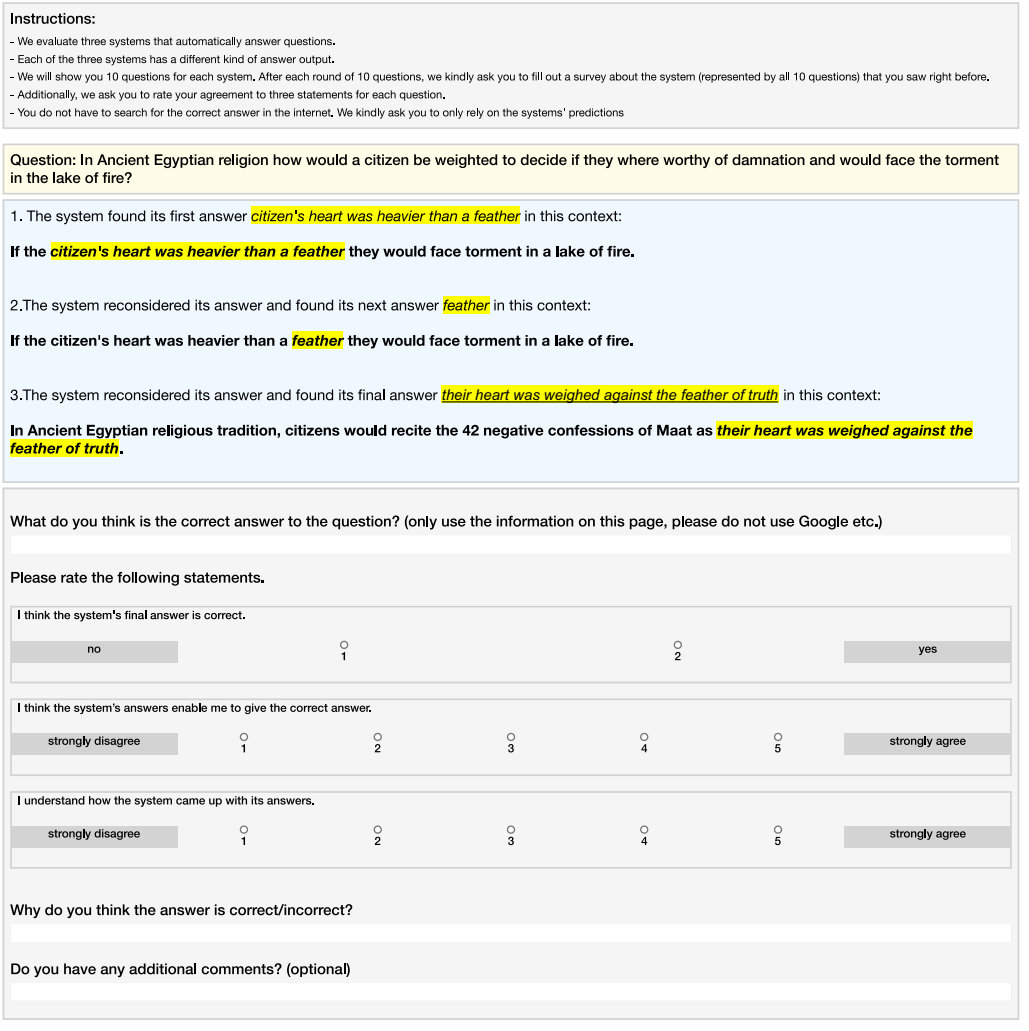


Figure 5. User study interface showing the TF condition (ours).

4.2.1 Statistical models

Per-System Ratings. To test for statistically significant effects of the choice of QA system (single answer, top-3, and thought flow) on the per-system ratings, we make use of Friedman tests (Pereira, Afonso, and Medeiros 2015) to account for the paired responses induced by the within-subject design.[o] We use Holm-corrected Conover post hoc tests to identify significant pairwise differences.[p]

Per-Item Ratings. Note that the within-subject design of our study possibly introduces interdependencies within ratings that we have to account for using an appropriate statistical model. Additionally, our dependent variables are measured on different levels, for example completion time is measured on a ratio scale while human answer correctness is measured on a nominal

[o]Although aggregated Likert item scores are commonly considered interval responses, we use commonly used (non-parametric) Friedman tests that only require ordinal responses and are more conservative than their parametric counterparts RM-ANOVAs.

[p]We refer to Schuff *et al.* (2023b) for an overview of applicable statistical tests.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table 7. Statistical results of our human evaluation ($N = 55$). "*" marks dependent variables on which a significant effect of the system condition was observed (Friedman tests and LRT tests for GLMM/CLMM). Pairwise differences between conditions (Holm-adjusted Tukey/Conover tests) are reported as compact letter display codings. For example, the "human-like" column shows that the post hoc test detected a significant difference between SINGLE and TF but no significant difference between any other pair. Similarly, the last column shows pairwise differences between all conditions and the TF condition reaches significantly higher human answer F_1-scores than any other conditions. Variables for which TF is among the best-performing models are marked in cyan, variables for which it is found to be the sole superior system are marked in green

	Perceived quality							User performance	
Condition	correct*	understand*	helpful*	usability	mental effort	humanlike*	intelligent*	time*	answer F1*
SINGLE	A	A	A	(A)	(A)	A	A	(A)	A
TOP-3	A	(B)	(B)	(A)	(A)	(AB)	(B)	B	B
TF	(B)	(B)	(B)	(A)	(A)	(B)	(B)	(AB)	(C)

Table 8. Detailed *p* values for all main effects and pairwise comparisons are shown in Table 7. Significant *p* values are marked in **bold**. Cell colors follow the color coding used in Table 7

	Perceived quality							User performance	
	correct*	understand*	helpful*	usability	mental effort	humanlike*	intelligent*	time*	answer F1*
Main effect	<**0.0001**	<**0.0001**	<**0.0001**	0.07968	0.6282	**0.03575**	**0.00124**	<**0.0001**	<**0.0001**
TF – SINGLE	<**0.0001**	<**0.0001**	<**0.0001**	0.13116	1	**0.03431**	**0.00586**	0.15304	<**0.0001**
TF – TOP-3	**0.00891**	0.8867	0.9994	0.84254	1	0.30556	1	0.06207	<**0.0001**
TOP-3 – SINGLE	0.51897	<**0.0001**	<**0.0001**	0.13653	1	0.25097	**0.00586**	**0.00012**	<**0.0001**

(dichotomous) scale.[q] We, therefore, use (generalized) linear mixed models (GLMM) and cumulative link mixed models (CLMM) to (i) account for random effects of question and subject IDs, and (ii) account for the variables' respective measurement scales.[r] We use Likelihood-ratio tests (LRT) between the full model and the model without the condition variable to identify main effects of the condition variable and conduct Holm-corrected Tukey post hoc tests.

4.2.2 Results

We find significant differences for all dependent variables except usability and mental effort. We summarize the results of our statistical analysis in Table 7 using CLD codings (Piepho 2004). Table 8 provides the *p* values for main effects and each pairwise comparison. In the following, we discuss our findings for each dependent variable for which we found a significant main effect.

Perceived Answer Correctness. While there is no statistically significant difference between showing single answers or top-3 predictions to users, displaying thought flows leads to significantly higher answer correctness ratings.

[q]We follow related work and treat Paas mental effort, UMUX, and Godpseed subscale responses as interval data but analyze single-item perceived understanding and helpfulness on an ordinal level.

[r]We use (G)LMMs to analyze continuous and dichotomous responses (Gamma/binomial link) and CLMMs to analyze ordinal ones.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Understanding. Top-3 as well as thought flow predictions significantly increased the subjectively perceived level of understanding of how the system came up with its answer compared to single predictions.

Helpfulness. Similarly, top-3 and the thought flow predictions significantly improve perceived system helpfulness compared to single predictions.

Anthropomorphism. While we observe no significant difference in anthropomorphism ratings between single and top-3 predictions, the thought flow predictions are perceived as significantly more human-like/natural than single answers.

Perceived intelligence. Both, top-3 and the thought flow predictions lead to significantly increased perceived system intelligence.

Completion Time. We observe that the top-3 predictions significantly improve completion times compared to single answers although there is no significant increase for thought flows.

User Performance. Compared to single answers, top-3 predictions already improve user performance in terms of F_1-score of the answer which the user decided on using the system. However, thought flow predictions allow for even higher human-AI performances which are significantly higher than answers given in the single answer or top-3 conditions. We additionally analyze user answers using exact match scores and observe the same effects and model orders.

Overall, our results indicate that **thought flows are better than or as good as single answer or top-3 predictions across all dimensions evaluated**. In particular for perceived answer correctness, humanlikeness, and user performance, thought flows are significantly better than both single answers and top-3 predictions. While comparable (statistically indistinguishable) improvements of understanding, helpfulness, naturalness, and intelligence can also be achieved using top-3 predictions, these come at the cost of significantly increased completion times compared to single answers. In contrast, we do not find a significant time increase using thought flows.

5. Application to other tasks and domains

So far, we explored our thought flow method in the context of QA systems. As our method only requires a model to provide a vector representation of the model input and a differentiably linked model output, it can be applied to the vast majority of, e.g., classification models within as well as outside NLP. In the following, we demonstrate an exemplary application to image classification.

We use a pre-trained vision Transformer model (Dosovitskiy *et al.* 2020) as base model and fine-tune the model on the CIFAR-10 and CIFAR-100 image classification datasets (Krizhevsky 2009). We use the ViT-L-32 model variant pre-trained on the ILSVRC-2012 ImageNet and the ImageNet-21k datasets (Deng *et al.* 2009) as described by Dosovitskiy et al. (2020).[s]

As for our QA implementation discussed in Section 2.3, we have to specify our choice of logit vector $\hat{\mathbf{z}}$, input representation $\phi(\mathbf{x})$, correctness score s, and correctness score prediction function f_{corr}. While our QA span extraction model did yield two probability distributions (one for the start position and one for the end position), we now only have to consider a single distribution over image classes. Following our notation in Section 2.3, we thus define $\hat{\mathbf{z}}$ to be the predicted class logits. As input representation $\phi(\mathbf{x})$, we use the vision Transformer's embedding of the [CLS] token as—in contrast to our QA model which used each token's embeddings—our image classifier only relies on the [CLS] embedding when predicting the image class. While we used F_1-score as correctness score in our QA experiments, we use a probability score s now, that is the correction module

[s]The models are available via https://github.com/google-research/vision_transformer.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

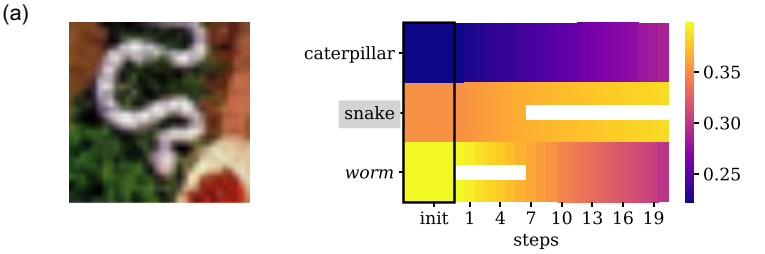


Figure 6. Exemplary thought flows on CIFAR-100 instances. The black rectangle shows the initial class probabilities from the base model (step 0), that is, the unmodified prediction, from a bird's eye perspective. The corresponding predicted label is marked in *italics*. On the right side of the black rectangle, the thought flow is depicted. The white lines mark the maximum probability across classes for each step. The ground-truth label is marked with a gray box. For readability, we only show classes that reach a probability of at least 1% within the thought flow.

predicts a probability estimate that the label prediction is correct.[t] As for our QA implementation, we implement f_{corr} as a two-layer multilayer perceptron (MLP) with scaled exponential linear unit (SELU) activation. We train the correction module using cross-entropy loss. Overall, we train five models for each of the datasets using different random seeds.

We observe that applying our thought flow can successfully correct erroneous predictions. Fig. 6 shows two examples. In Fig. 6(a), the wrong prediction *worm* is corrected to *snake* after eight gradient steps. Similarly, Fig. 6(b) shows a correction from *forest* to *bridge*. While the probability mass is redistributed over the course of the thought flow, the class *road* gains probability as well which can be interpreted as a sensible "change of mind" as the central object could be a road on a bridge as well.

In terms of accuracy, our models yield consistent but small performance gains (<0.3% for both datasets). However, as our baseline models reach 98.7% (SD = 0.7) accuracy on CIFAR-10 and 92.5% (SD = 0.7) accuracy on CIFAR-100, there is much less room for improvement than in our QA experiments for which our base model reached 63.5% F_1-score.

We provide a detailed analysis of thought flow patterns for image classification similar to our analysis for QA in Section 3.3 in Appendix D.

Overall, we observe that our thought flow method is applicable beyond QA and can correct model predictions of image classifiers. As for the QA thought flow patterns discussed in Section 3.3, we observe numerous correction patterns that exhibit a surprisingly high complexity and motivate a deeper study of the correction dynamics in future work.

[t]We also experimented with predicting true-class probability instead of correctness probability, similar to Corbière et al. (2019), but did not observe improvements over our setting.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

6. Related work

6.1 Cognitive modeling and systems

The fields of cognitive modeling and cognitive systems have developed and evaluated numerous theories and models of human thinking (Rupert 2009; Busemeyer and Diederich 2010; Levine 2018; Lake *et al*. 2017). While work in these fields is often oriented towards accurate descriptions of human cognition, our method does not aim to provide a plausible description of cognitive process *per se*. We instead aim to apply a mature, well-studied philosophical concept to machine learning in order to improve classification performance and user utility.

6.2 Confidence estimation

Of the existing methods to estimate a model's confidence and the correctness of its predictions, the broader approach which trains secondary models for predicting the main model's uncertainty (e.g., Blatz *et al.* 2004; DeVries and Taylor 2018) is the closest to our work. *ConfidNet* (Corbière *et al*. 2019) is particularly relevant to our approach as it predicts the true-class probability of the main model. In contrast, our correction module receives the class probabilities of the main model as input and predicts a correctness score. Unlike methods which aim at estimating accurate confidence scores, we predict such scores only as an auxiliary task in order to generate a gradient that allows us to update the model prediction.

6.3 Model corrections

Regarding model correction, the arguably most established approach to learning corrections of model predictions is gradient boosting (Friedman 2001) including its popular variant XGBoost (Chen and Guestrin 2016). Unlike these methods which aim at using an ensemble of weak learners, we propose a lightweight correction module that is applicable on top of any existing classification model. Furthermore, in our method, the correction module receives the main model's predictions and is able to directly adapt them.

6.4 Sequences of predictions

The idea of iteratively predicting and correcting model responses has been explored for a long time. Early work includes Mori et al. who present a non-neural iterative correction method tailored to estimate elevation maps from aerial stereo imagery (Mori, Kidode, and Asada 1973). Katupitiya et al. propose to iterate two neural networks to address the problem of predicting inputs of a mechanical process given the outputs of the process (Katupitiya and Gock 2005). While their method is specifically designed for the task of input prediction, our work presents a general-purpose classification model that iterates class label predictions.

Besides those task-specific methods, there are models and inference methods that make use of an iterative prediction process by design, such as Hopfield networks (Hopfield 1982) and their modern variants (Barra, Beccaria, and Fachechi 2018; Ramsauer *et al*. 2020), or Loopy Belief Propagation, Markov Chain Monte Carlo or Gibbs sampling (Bishop 2006; Koller and Friedman 2009). While these techniques can be linked to our work conceptually, they all require a new model to be trained. In contrast, our approach can be applied to an existing neural model as well.

6.5 Chain-of-thought, tree-of-thoughts, and self-refine

Another related line of work uses particular prompting strategies to improve the output of language models.

In chain-of-thought (CoT) prompting (Wei and Wang 2022), the language model is prompted with examples of expected answers, correct/incorrect examples, problem decomposition, or reasoning in a few-shot or one-shot manner which are likely to bias, condition, and ground

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

the large language model in its responses. A typical zero-shot variant of CoT prompting is, for example, to append "Let's think step by step" to the prompt (Kojima *et al.* 2022).

While CoT prompting methods can yield improved model responses, they (a) typically predict only one answer which provides an opaque view of the models' deduction and reasoning steps without changing or correcting the answer itself and (b) are restricted to a textual input domain. In contrast, our method is applicable to any domain that can be transformed into a vector representation such as images, sound, or graphs. In addition, while CoT prompting yields *one* answer that contains information on its deduction without changing or correcting its answer (i.e., it cannot be applied iteratively), our method is not specifically targeted towards decomposition/reasoning in that it predicts a sequence of answers towards the goal of iteratively improving the answer with every iteration.

As a generalization of chain-of-thought-prompting, tree-of-thoughts prompting (Yao *et al.* 2023) explores sampled candidate solutions using a heuristically guided tree search. Similar to our approach, this enables iterative decision updates. However, tree-of-thought-prompting is limited to textual inputs and can only represent discrete decision changes instead of the smooth solution space exploration provided by our method. We refer to the survey of Yu *et al.* (2023) for an exhaustive taxonomy of CoT prompting strategies.

Similarly, Madaan *et al.* (2023) propose Self-Refine, a prompt-based method that is used to iteratively generate an initial response, textual feedback to that response, and a subsequently updated response. While Self-Refine and our method share the paradigm of iterative self-critique and response revision, Self-Refine is constrained to the textual domain and requires the underlying model to be a text generation model that supports the method's feedback and revision prompts, while our method can be applied to arbitrary domains and supports any type of model. This difference to our method equally applies to numerous other variants of Self-Refine such as Self-Correct (Welleck *et al.* 2023) and Reflexion (Shinn *et al.* 2023).

Building upon the insights reported by Dai *et al.* (2023), who show that in-context learning can have a similar effect as explicit finetuning, we hypothesize that the described prompt-based methods can potentially be linked to our method via the approximately dual form between Transformer attention and gradient descent.

6.6 Learning to stop

In another related line of work, namely ACT (Graves 2016) and PonderNet (Banino, Balaguer, and Blundell 2021) recurrent networks are trained to learn when to stop applying recurrent transformations within the model. While their approaches call for recurrent modules and base model retraining, our method only requires the model to yield output logits and leaves the base model unchanged.

7. Limitations and future work

The previous results demonstrate that our thought flow method can improve model performance (Section 3.2) as well as users' system perception and human-AI performance (Section 4.2). However, its successful application to improve model performance is yet bound to limitations. The reasons for its utility for users needs to be systematically explored, and its potential to be applied to a broad range of domains and tasks should be exploited. In the following, we detail these three aspects and propose concrete directions for future work.

7.1 Stopping heuristics

This paper conducts an initial exploration of the thought flow method's feasibility and demonstrates that the proposed training and inference algorithms can successfully be applied to a

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

complex QA task and on top of the prevalent Transformer architecture. However, our experiments in Section 3.2 also show that a crucial factor of its successful application is to stop prediction modifications (a) late enough to enable class label changes but (b) early enough to prevent "overshooting" modifications that, despite an initial prediction improvement, deteriorate the prediction in later modifications. We hypothesize that this challenge can be successfully tackled by heuristically exploiting (i) geometric features of the thought flow (e.g., smooth transitions could potentially signal desirable prediction modifications while sharp and sudden changes could signal overshooting), (ii) decaying gains in self-estimated correctness (e.g., as soon as the model's correctness score estimates improvements per prediction update fall below a threshold, the corresponding prediction updates might not yield additional improvements), or (iii) considering thought flows as dynamic systems and deriving stopping criteria from established characteristics. Concretely, we argue that the study of curvature can be a promising first step for geometry-related stopping heuristics, a running average over self-estimated correctness differences between iterations can be applied for correctness score-related stopping heuristics, and Lyapunov exponents (Dieci and Van Vleck 2002) should be studied for stopping heuristics considering thought flows as dynamic systems. We expect that future work will be able to develop stopping heuristics by exploring these directions either in isolation or in combination and thereby enable real-world deployment of the thought flow method.

7.2 Underlying effects on human perception

Our results in Section 4.2 show that thought flows are better or equally good than both (a) single predictions and (b) top-3 predictions. While many of the improvements of thought flows over single predictions can also be achieved by providing users with top-3 predictions, top-3 predictions induce a cost of significantly increased completion times. These observations raise the question of how thought flows differ from top-3 predictions regarding human perception such that they enable equal or better user interaction. We expect that answering this question requires to explore how thought flow *versus* top-3 predictions can be related to humans' processing of sequential *versus* simultaneous presentation. Concretely, we hypothesize thought flows to be associated with sequential presentation and top-3 predictions with simultaneous presentation. Related work in psychology found simultaneous presentation to be associated with a focus on the difference between options and an attention shift away from common information (Basu and Savani 2018) and—in specific conditions—worse decisions than sequential presentation (Read *et al.* 2001). Future work should explore this potential relation between thought flows and users' processing of sequential presentation.

7.3 Applicability to further tasks and domains

In this paper, we introduced our thought flow method with the example of a QA task in the text domain. However, as it only requires a model to provide a vector representation of the model input and a differentiably linked model output, it can be applied to the vast majority of classification models within as well as outside natural language processing. In general, we expect that our method will yield promising results in any task for which predicting the correct answer is substantially harder than verifying whether a given answer is plausible. We thus hypothesize that our method will be particularly useful for tasks that involve (a) multiple reasoning steps, (b) structured predictions, or (c) generation. Beyond the multi-step QA task we explored, multi-step reasoning tasks in NLP include multi-step reading comprehension (Lin *et al.* 2019) or multi-step numerical reasoning (Zhao *et al.* 2022). Structured prediction tasks in NLP include, i.a., dependency parsing or multi-label document classification. A concrete example of a generation task is dialog systems for which our flow method could prevent systems from generating inconsistent responses by reassessing their answer generation with respect to the dialog history. In addition, our method is

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

agnostic to the input domain, motivating various applications in the vision domain, such as scene comprehension or image segmentation.

8. Conclusion

In this paper, we introduced a task-agnostic self-correction formalism that turns a model's single output prediction into a "*thought flow*"—an evolving sequence of predictions. We draw inspiration from Hegel's dialectics and propose a correction module along with a gradient-based update rule that sequentially updates a model's output distributions in the direction of an increasing self-estimate of correctness. We incorporate our method into existing QA models and conduct extensive experiments which include downstream human evaluation. Our experimental results indicate that thought flows (i) can increase F_1-scores by up to 9.3%, (ii) exhibit complex self-modification patterns, (iii) provide significant improvements in human interaction and system perception including task performance as measured in task performance and perceived system correctness and naturalness, and (iv) can be applied to further domains, such as computer vision. A potential next step to further improve performance is a mechanism to learn when to stop the modification procedure for which we discuss different heuristic approaches.

Competing interests. The authors declare none.

References

Banino A., Balaguer J. and Blundell C. (2021). Pondernet: Learning to ponder. CoRR, abs/2107.05407.

Barra A., Beccaria M. and Fachechi A. (2018). A new mechanical approach to handle generalized Hopfield neural networks. *Neural Networks* **106**, 205–222.

Bartneck C., Kulic D., Croft E.A. and Zoghbi S. (2009). Measurement instruments for the anthropomorphism, animacy, likeability, perceived intelligence, and perceived safety of robots. *International Journal of Social Robotics* **1**(1), 71–81.

Basu S. and Savani K. (2018). Choosing among options presented sequentially or simultaneously. *Current Directions in Psychological Science* **28**, 97–101.

Beltagy I., Peters M.E. and Cohan A. (2020). Longformer: The Long-Document Transformer. CoRR, abs/2004.05150. arXiv: 2004.05150.

Bishop C.M. (2006). *Pattern Recognition and Machine Learning*. New York, USA: Springer.

Blatz J., Fitzgerald E., Foster G., Gandrabur S., Goutte C., Kulesza A., Sanchis A. and Ueffing N. (2004). Confidence estimation for machine translation. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland. COLING, pp. 315–321.

Boonstra E.A. and Slagter H.A. (2019). The dialectics of free energy minimization. *Frontiers in Systems Neuroscience* **13**, 42. Publisher: Frontiers.

Busemeyer J.R. and Diederich A. (2010). *Cognitive Modeling*. Thousand Oaks, CA, USA: Sage.

Chen T. and Guestrin C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 13-17, 2016, San Francisco, CA, USA. ACM, pp. 785–794.

Corbière C., Thome N., Bar-Hen A., Cord M. and Pérez P. (2019). Addressing failure prediction by learning model confidence. In Wallach H., Larochelle H., Beygelzimer A., d' Alché-Buc F., Fox E. and Garnett R., (eds), *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc.

Dai D., Sun Y., Dong L., Hao Y., Ma S., Sui Z. and Wei F. (2023). Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In Rogers A., Boyd-Graber J. L. and Okazaki N. (eds), *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*. Association for Computational Linguistics, pp. 4005–4019.

Deng J., Dong W., Socher R., Li L.-J., Li K. and Fei-Fei L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009)*, Miami, Florida, USA. IEEE Computer Society, pp. 248–255.

Devlin J., Chang M.-W., Lee K. and Toutanova K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota. Association for Computational Linguistics, pp. 4171–4186.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

DeVries T. and Taylor G.W. (2018). Learning Confidence for Out-of-Distribution Detection in Neural Networks. CoRR, abs/1802.04865. _eprint: 1802.04865.

Dieci L. and Van Vleck E.S. (2002). Lyapunov and other spectra: a survey. In *Collected Lectures on the Preservation of Stability under Discretization (Fort Collins, CO, 2001)*, pp. 197–218.

Dosovitskiy A., Beyer L., Kolesnikov A., Weissenborn D., Zhai X., Unterthiner T., Dehghani M., Minderer M., Heigold G., Gelly S., Uszkoreit J. and Houlsby N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. CoRR, abs/2010.11929. _eprint: 2010.11929.

Ficara E. and Priest G. (2023). Introduction: The formalization of dialectics. *History and Philosophy of Logic* **44**(2), 115–118.

Finstad K. (2010). The usability metric for user experience. *Interacting with Computers* **22**(5), 323–327.

Finstad K. (2013). Response to commentaries on 'the usability metric for user experience'. *Interacting with Computers* **25**(4), 327–330.

Friedman J.H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics* **29**(5), 1189–1232.

Gal Y. and Ghahramani Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016*, June 19-24, 2016, New York City, NY, USA, vol. 48, pp. 1050–1059. JMLR Workshop and Conference Proceedings, JMLR.org.

Graves A. (2016). Adaptive computation time for recurrent neural networks. CoRR, abs/1603.08983.

Guo C., Pleiss G., Sun Y. and Weinberger K.Q. (2017). On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, 6-11 August 2017, Sydney, NSW, Australia, vol. 70, pp. 1321–1330, Proceedings of Machine Learning Research, PMLR.

Hopfield J.J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* **79**(8), 2554–2558. Publisher: National Acad Sciences.

Kadioglu S. and Sellmann M. (2009). Dialectic search. In *International Conference on Principles and Practice of Constraint Programming*. Springer, pp. 486–500.

Katupitiya J. and Gock K. (2005). Neural network based iterative prediction of multivariable processes. In *IEEE International Conference Mechatronics and Automation*. IEEE, vol. 4, pp. 2043–2048.

Kaya Y., Hong S. and Dumitras T. (2019). Shallow-deep networks: Understanding and mitigating network overthinking. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, Long Beach, California, USA, vol. 97, pp. 3301–3310, Proceedings of Machine Learning Research, PMLR.

Klambauer G., Unterthiner T., Mayr A. and Hochreiter S. (2017). Self-normalizing neural networks. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017*, December 4-9, 2017, Long Beach, CA, USA, pp., 971–980.

Kojima T., Gu S.S., Reid M., Matsuo Y. and Iwasawa Y. (2022). Large language models are zero-shot reasoners. In *NeurIPS*.

Koller D. and Friedman N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, Massachusetts, USA: MIT Press.

Krizhevsky A. (2009). Learning Multiple Layers of Features from Tiny Images.

Kuçuradi I. (1990). Different concepts of dialectics: Methods and views. *Studies in Soviet Thought* **39**(3-4), 257–264.

Lake B.M., Ullman T.D., Tenenbaum J.B. and Gershman S.J. (2017). Building machines that learn and think like people. *Behavioral and Brain Sciences* **40**. Publisher: Cambridge University Press.

Levine D.S. (2018). *Introduction to Neural and Cognitive Modeling*. Oxfordshire, UK: Routledge.

Lin K., Tafjord O., Clark P. and Gardner M. (2019). Reasoning over paragraph effects in situations. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, Hong Kong, China. Association for Computational Linguistics, pp. 58–62.

Loshchilov I. and Hutter F. (2019). Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*, May 6-9, 2019, New Orleans, LA, USA. OpenReview.net.

Madaan A., Tandon N., Gupta P., Hallinan S., Gao L., Wiegreffe S., Alon U., Dziri N., Prabhumoye S., Yang Y., Gupta S., Majumder B.P., Hermann K., Welleck S., Yazdanbakhsh A. and Clark P. (2023). Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, December 10-16, 2023, New Orleans, LA, USA.

Maybee J.E. (2020). Hegel's dialectics. In Zalta E.N., (ed), *The Stanford Encyclopedia of Philosophy*, winter 2020 edition, Metaphysics Research Lab, Stanford University.

Mori K.-i., Kidode M. and Asada H. (1973). An iterative prediction and correction method for automatic stereocomparison. *Computer Vision, Graphics, and Image Processing* **2**(3-4), 393–401.

Mueller G.E. (1958). The hegel legend of "thesis-antithesis-synthesis". *Journal of the History of Ideas* **19**(3), 411–414. Publisher: JSTOR.

Nuzzo A. (2023). Form, formality, formalism in hegel's dialectic-speculative logic. *History and Philosophy of Logic* **44**(2), 169–183.

Paas F.G.W.C. (1992). Training strategies for attaining transfer of problem-solving skill in statistics: a cognitive-load approach. *Journal of Educational Psychology* **84**(4), 429–434.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Pereira D.G., Afonso A. and Medeiros F.M. (2015). Overview of friedman's test and post-hoc analysis. *Communications in Statistics - Simulation and Computation* **44**(10), 2636–2653.
Piepho H.-P. (2004). An algorithm for a letter-based representation of all-pairwise comparisons. *Journal of Computational and Graphical Statistics* **13**(2), 456–466.
Priest G. (2023). The logical structure of dialectic. *History and Philosophy of Logic* **44**(2), 200–208.
Ramsauer H., Schäfl B., Lehner J., Seidl P., Widrich M., Gruber L., Holzleitner M., Pavlovic M., Sandve G.K., Greiff V., Kreil D.P., Kopp M., Klambauer G., Brandstetter J. and Hochreiter S. (2020). Hopfield Networks is All You Need. CoRR, abs/2008.02217. _eprint: 2008.02217.
Read D., Antonides G., Van den Ouden L. and Trienekens H. (2001). Which is better: Simultaneous or sequential choice? *Organizational Behavior and Human Decision Processes* **84**(1), 54–70.
Riegel K.F. (1973). Dialectic operations: The final period of cognitive development. *Human Development* **16**(5), 346–370.
Rupert R.D. (2009). *Cognitive Systems and the Extended Mind*. Oxford, UK: Oxford University Press.
Schuff H., Adel H., Qi P. and Vu N.T. (2023). Challenges in explanation quality evaluation. CoRR, abs/2210.07126.
Schuff H., Vanderlyn L., Adel H. and Vu N.T. (2023). How to do human evaluation: A brief introduction to user studies in nlp. *Natural Language Engineering* **29**(5), 1–24.
Shinn N., Cassano F., Gopinath A., Narasimhan K. and Yao S. (2023). Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, December 10-16, 2023, New Orleans, LA, USA.
Wei J. and Wang X. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems 35: Annual Conferenceon Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022.
Welleck S., Lu X., West P., Brahman F., Shen T., Khashabi D. and Choi Y. (2023). Generating sequences by learning to self-correct. In *The Eleventh International Conference on Learning Representations, ICLR 2023*, May 1-5, 2023, Kigali, Rwanda. OpenReview.net.
Yang Z., Qi P., Zhang S., Bengio Y., Cohen W., Salakhutdinov R. and Manning C.D. (2018). HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics, pp. 2369–2380.
Yao S., Yu D., Zhao J., Shafran I., Griffiths T.L., Cao Y. and Narasimhan K. (2023). Tree of thoughts: Deliberate problem solving with large language models. CoRR, abs/2305.10601.
Yu Z., He L., Wu Z., Dai X. and Chen J. (2023). Towards better chain-of-thought prompting strategies: A survey. CoRR, abs/2310.04959.
Zhao Y., Li Y., Li C. and Zhang R. (2022). MultiHiertt: Numerical reasoning over multi hierarchical tabular and textual data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland. Association for Computational Linguistics, pp. 6588–6600.

Appendix A. Relation to Hegel's dialectics

Dialectics, in general, refers to a family of methods pertaining to philosophical argumentation that involves contradictory processes between opposing sides (Maybee 2020). What distinguishes Hegel's dialectic from other dialectics is that in his dialectic the opposing sides are views or definitions while the most classic version, Plato's dialectic, treats people as the opposing sides, for example Maybee (2020). While there is a variety of dialectics that can potentially inspire the development of novel machine learning methods (see, e.g., the overview of dialectics by Kuçuradi 1990), we choose to build upon Hegel's dialectics due to its philosophical relevance and maturity as well as its relation to various fields, such as cognitive sciences (Riegel 1973), neuroscience (Boonstra and Slagter 2019), or constraint satisfaction and optimization (Kadioglu and Sellmann 2009).

A.1 Relation to thesis-antithesis-synthesis triads

The three moments are often compared to the thesis-antithesis-synthesis pattern in philosophy popularized by Heinrich Moritz Chalybäus, but are not necessarily equal (Mueller 1958). A more detailed discussion of the thesis-antithesis-synthesis triad and its difference to the three moments is provided by Maybee (2020). While the thesis-antithesis-synthesis triad suggests a singular dialectical "one-pass" process of thinking and reasoning, Hegel's dialectical process allows for several iterations and does not have to end after a single iteration (Maybee

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

2020). A particular example of such an iterative process within Hegel's work can be found in the dialectical development of Hegel's logic regarding the concepts of "Abstract Purpose" and "Realized Purpose" (Maybee 2020). The fundamental "multi-pass" notion of multiple iterations of thinking and reasoning forms the basis for of our thought flow approach.

Appendix B. Question answering experiments

B.1 Data set details

We use the HOTPOTQA data set (Yang *et al.* 2018), which is an English multi-hop QA data set. It covers 90,564 training instances, 7,405 test validation instances, and 7,405 test instances per setting (there are a distractor and a fullwiki setting). Training instances are grouped by difficulty and cover 18,089 easy, 56,814 medium, and 15,661 hard questions. We refer to Yang *et al.* (2018) for more details.

B.2 Additional modification examples

Table B1 displays additional examples for each of the emerging thought flow patterns discussed in Section 3.3.

B.3 Changes in precision and recall

Figure B1 shows the fractions of instances with an observed increase or decrease in precision and recall across different δ values.

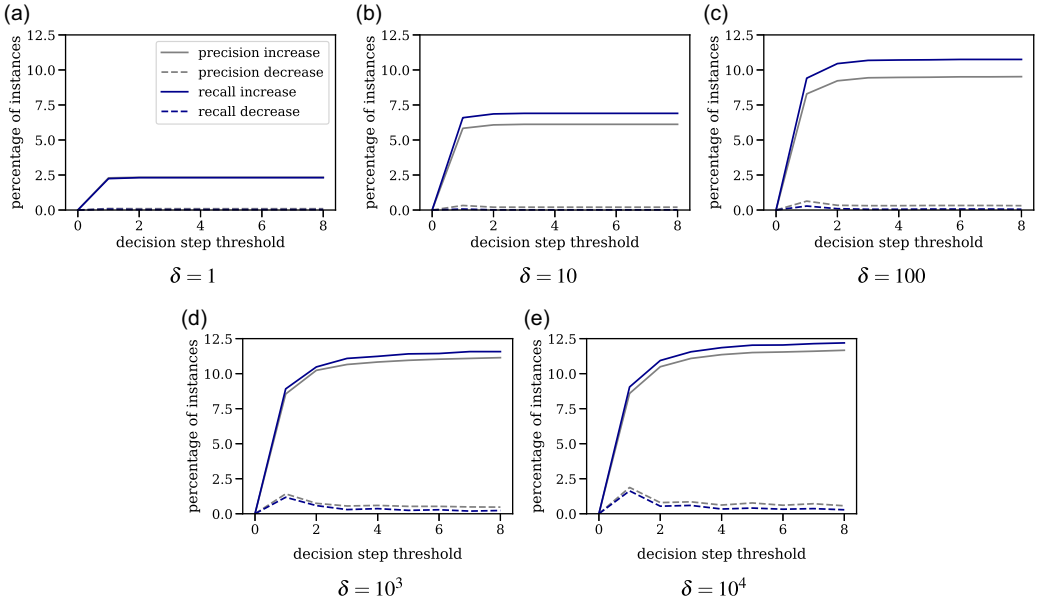


Figure B1. Fraction of instances for which a thought flow with different decision step thresholds results in increased or decreased values of precision and recall.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Table B1. Additional examples of the modification patterns presented in Table 2. The **correct answer** is marked in bold, and the predicted answer per flow step is marked in orange

Pattern	Example
cross-sentence	Question: How many novels are there in series of novels which were adapted for a 2015 BBC TV series in which Ruby Bentall plays a character called Verity ?
	(1) The Verity Birdwood series is a series of six murder mystery novels by Jennifer Rowe
	(2) The series comprises **12** novels: the first seven are set in the 18th century, concluding in Christmas 1799; the remaining five are concerned with the early years of the 19th century and the lives of the descendants of the previous novels' main characters
reduction	Question: Who was one of the stars who played the two oldest children in a TV series which had a 2010 movie based on it?
	(1) The series stars **Michael Seater** and Ashley Leggat as the two oldest children in a stepfamily
	(2) The series stars **Michael Seater** and Ashley Leggat as the two oldest children in a stepfamily
extension	The Admiral's Men occupied which kind of theatre in the 1590s?
	(1) The Rose was an **Elizabethan** **theatre**
	(2) The Rose was an **Elizabethan theatre**
in-sentence	Question: The song "Only a Pawn in Their Game" was based on a crime whose perpetrator was convicted in this year
	(1) Byron De La Beckwith, Sr. (November 9, 1920 - January 21, 2001) was an American white supremacist and Klansman from Greenwood, Mississippi, who in **1994** was convicted of assassinating civil rights leader Medgar Wiley Evers on June 12, 1963
	(2) Byron De La Beckwith, Sr. (November 9, 1920 - January 21, 2001) was an American white supremacist and Klansman from Greenwood, Mississippi, who in **1994** was convicted of assassinating civil rights leader Medgar Wiley Evers on June 12, 1963
entity refinement	Question: How many Grammy Award nominations has the musician who Graham Maby has recorded and toured with since his first album, and whose first release was "Is She Really Going Out with Him," won?
	(1) He is a **5**-time Grammy Award winner and has participated in the making of numerous albums that have resulted in Grammy Award nominations and winners
	(2) He has recorded 19 studio albums and won **5** Grammy Award nominations throughout the course of his career
logic hops	Question: What position in the court does the Professor of Law at the Interdisciplinary Center in Herzliya hold?
	(1) Aharon Barak (Hebrew: Ahrn brk, born Aharon Brick, 16 September 1936) is a Professor of Law at the Interdisciplinary Center in Herzliya and a lecturer in law at the Hebrew University of Jerusalem, the Yale Law School, Central European University, Georgetown University Law Center, and the University of Toronto Faculty of Law
	(2) Its other two members were **Supreme Court Judge** Aharon Barak, and Major general (res.)
combined	Question: What American agronomist that worked on The Green Revolution won a Nobel Peace Prize?
	(1) Norman Ernest Borlaug (March 25, 1914September 12, 2009) was an American agronomist and humanitarian who led initiatives worldwide that contributed to the extensive increases in agricultural production termed the Green Revolution
	(2) The main development was higher-yielding varieties of wheat, which were developed by many scientists, including Indian geneticist M. S. Swaminathan, American agronomist Dr. **Norman Borlaug** , and others

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

Appendix C. User study

C.1 Questionnaire items

C.1.1 Per-system questionnaires

Usability. The UMUX usability scale (Finstad 2010, 2013) uses the following four 5-point Likert items:

- This system's capabilities meet my requirements.
- Using this system is a frustrating experience.
- This system is easy to use.
- I have to spend too much time correcting things with this system.

Mental Effort. The Pass mental effort scale usability scale (Paas 1992) uses a single 9-point Likert item:

- Please rate the mental effort required to decide if the system's answer is correct. (The 9 points are labeled from "very, very low mental effort" to "very, very high mental effort.")

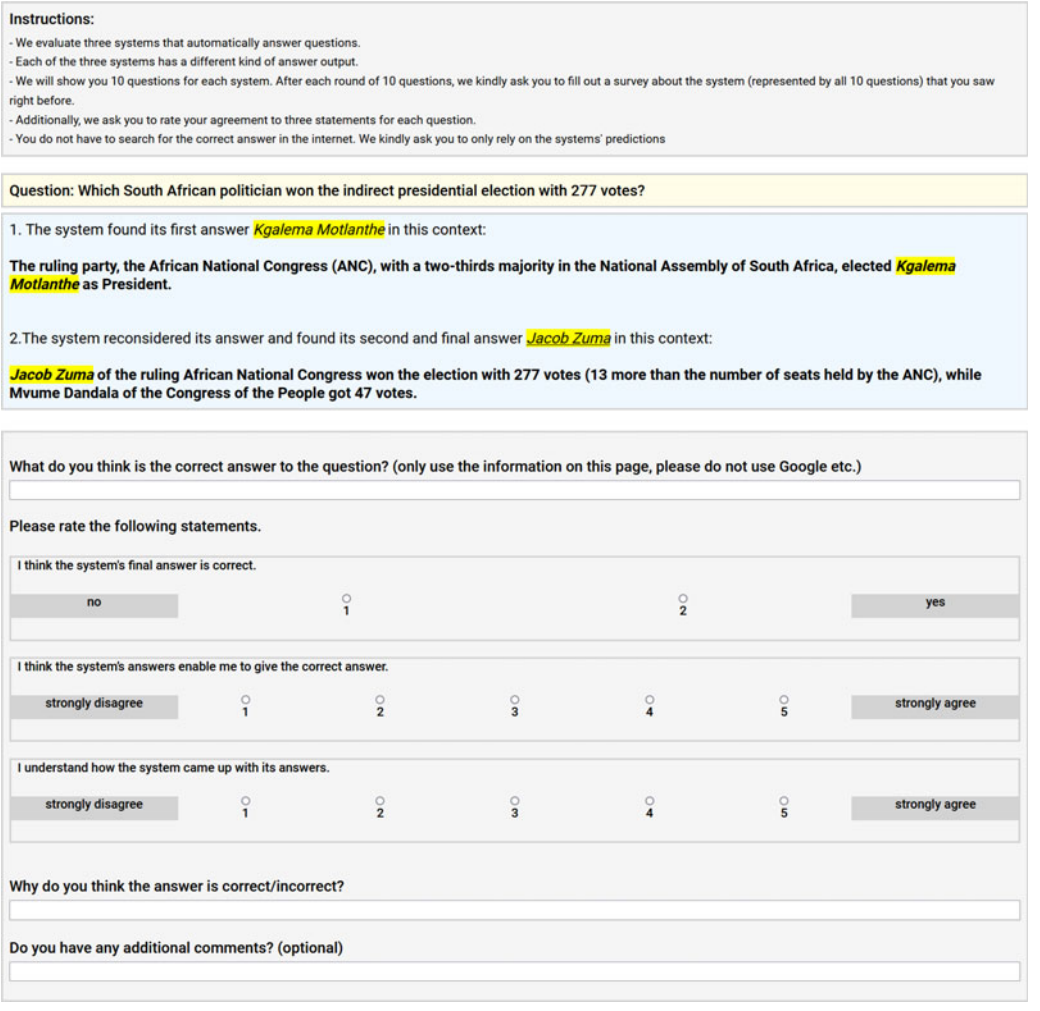


Figure C1. User study interface showing the TF condition (ours).

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

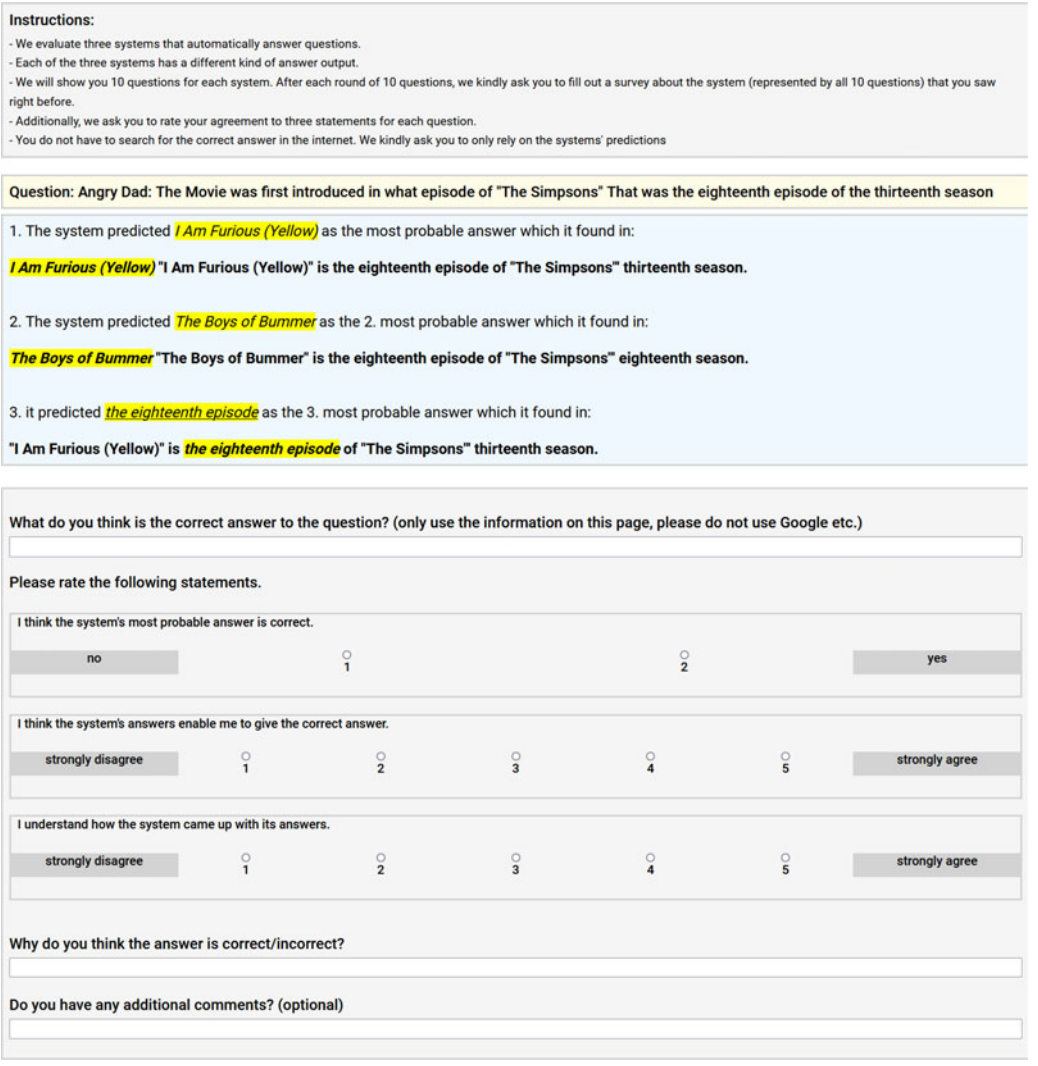


Figure C2. User study interface showing the TOP-3 condition.

Anthropomorphism. The Godspeed anthropomorphism subscale (Bartneck *et al.* 2009) uses five 5-point semantic differential scales that ask the user to rate the system in a spectrum of:

- fake – natural
- machine-like – human-like
- unconscious – conscious
- artificial – lifelike
- (moving rigidly – moving elegantly) (We exclude this item as it is not applicable to QA systems.)

Perceived Intelligence. The Godspeed perceived Intelligence subscale (Bartneck *et al.* 2009) uses five 5-point semantic differential scales that ask the user to rate the system in a spectrum of:

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

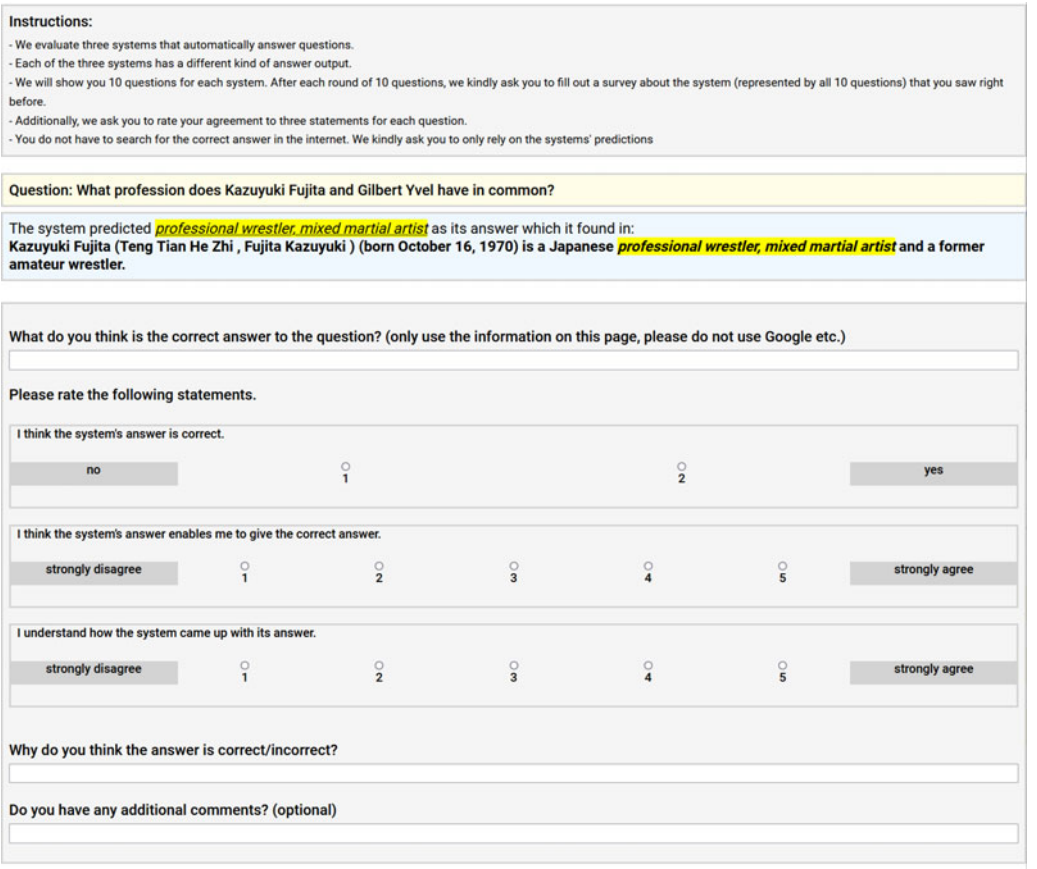


Figure C3. User study interface showing the SINGLE condition.

- incompetent – competent
- ignorant – knowledgeable
- irresponsible – responsible
- unintelligent – intelligent
- foolish – sensible

C.1.2 Per-item questionnaires

Perceived Answer Correctness. We use a single binary item to collect perceived answer correctness ratings:

- I think the system's answer is correct.

Perceived Helpfulness. We use a single 5-point Likert item to collect helpfulness ratings:

- I think the system's answer enables me to give the correct answer.

Perceived Understanding. We use a single 5-point Likert item to collect understanding ratings:

- I understand how the system came up with its answer.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

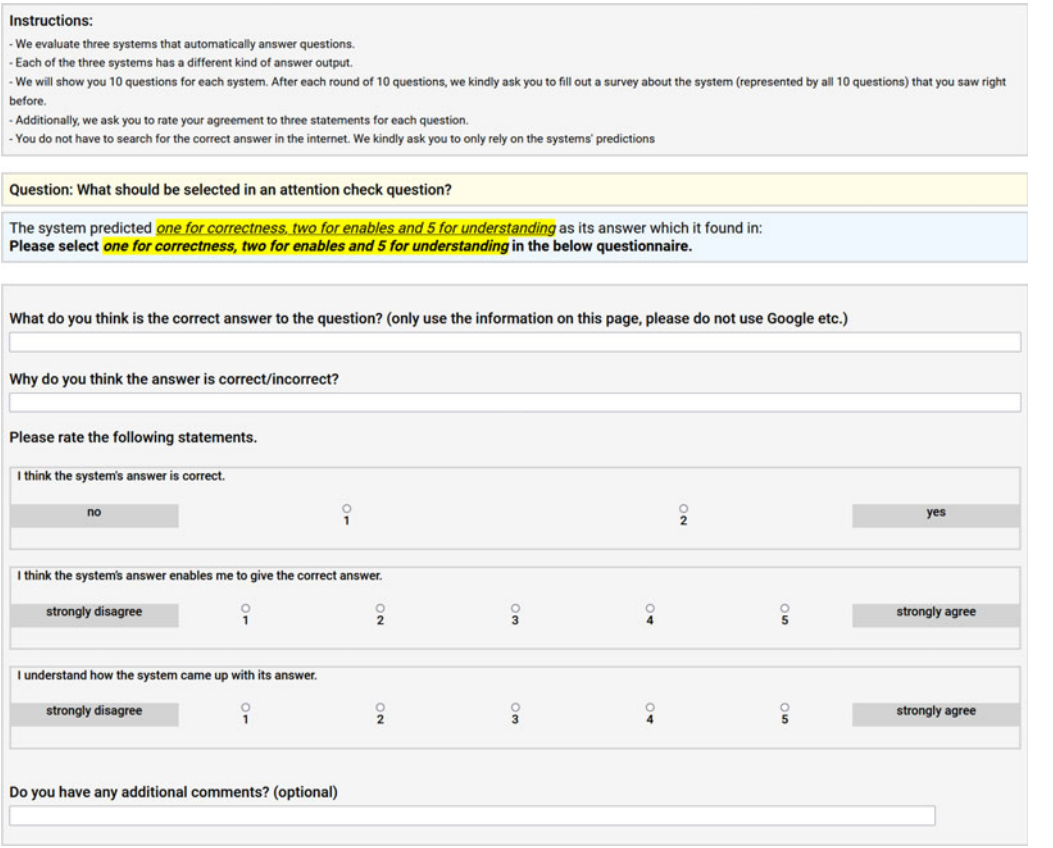


Figure C4. User study interface showing an attention check.

C.2 Interface

Figures C1, C2, C3 show screenshots of our experiment interface for the three studied prediction conditions TF, TOP-3 and SINGLE. Figure C4 depicts an attention check question.

Appendix D. Image classification thought flow patterns

Similar to the qualitative analysis of flow patterns in our QA experiments (see Section 3.3), we now investigate the dynamics of the generated image classification thought flows. While Fig. 6 shows thought flows that gradually transition from one class to another and then converge to that class, we observe diverse flow patterns which we display in Figs. D1 and D2. Fig. D1(a) shows an example for which our method does not change the (correct) label prediction but increases the model's confidence in its prediction. In Fig. D1(b), the thought flow does not change the predicted label but decreases the model's confidence. Thus, the flows in Fig. D1(a),(b) can be interpreted as a form of neural network calibration (Guo *et al.* 2017). Fig. D1(d) shows a smooth transition from one class to a gradual back-and-forth between two classes. In Fig. D1(e) one can see a transition from the class *tulip* to the class *sweet pepper* via the class *poppy*. In Fig. D2(a), the thought flow quickly changes from *plain* to *cloud*. While the predicted class remains *cloud*, the probability of plain decreases continuously until the flow changes its prediction to *sea* which we interpret as *overthinking* (Kaya, Hong, and Dumitras 2019). Fig. D1(b),(c),(c),(d) show different

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

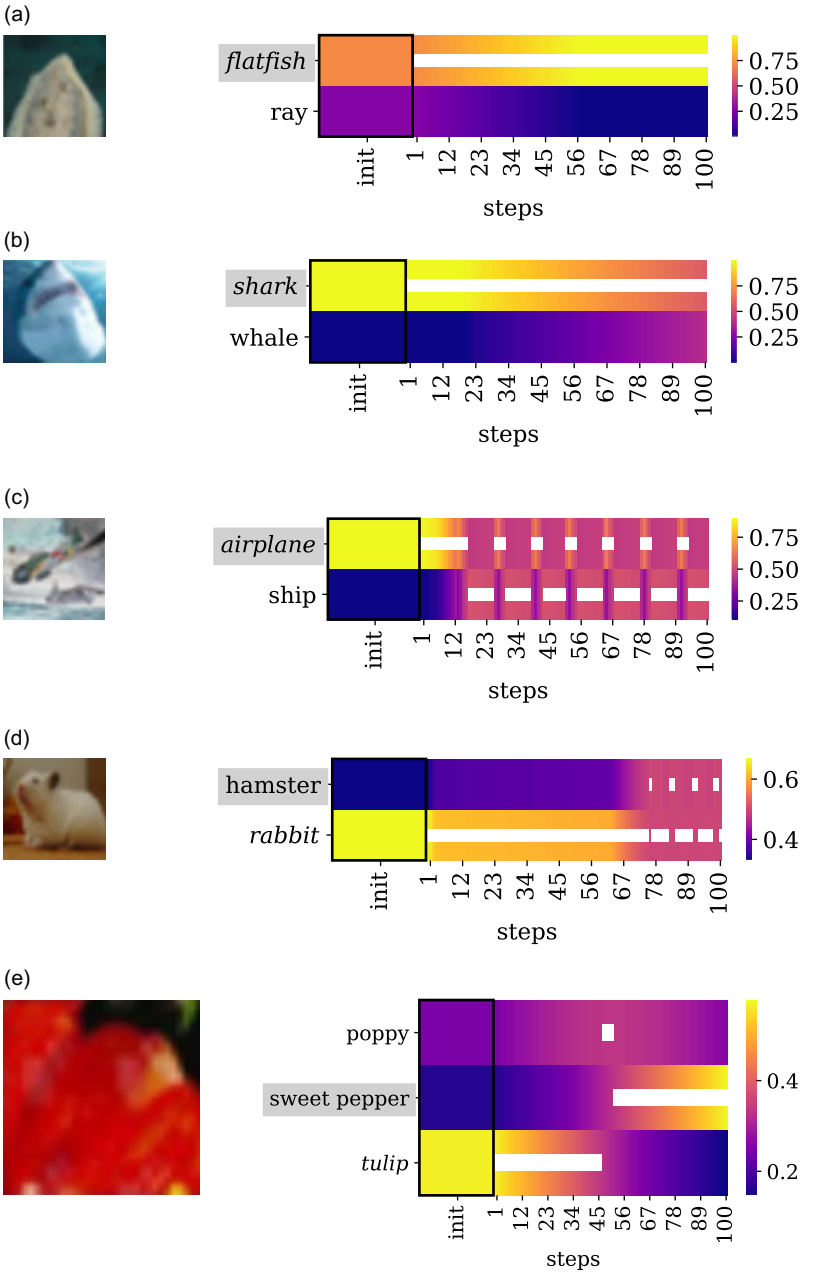


Figure D1. Exemplary thought flows from different models on CIFAR demonstrating the diverse range of correction dynamics. A detailed description of the plots is provided in Fig. 6.

periodic behaviors including the transition from a cycle to a fixed class in Fig. D2(b), smooth cycles in Fig. D1(c) and longer sawtooth-like cycles in Fig. D2(d). Importantly, Fig. D2(b), (e) are examples for flows that explore an alternative class prediction but "return" to the initial class prediction and thus show that our method can be used to explore alternatives without necessarily neglecting a correct prediction.

https://doi.org/10.1017/nlp.2024.41 Published online by Cambridge University Press

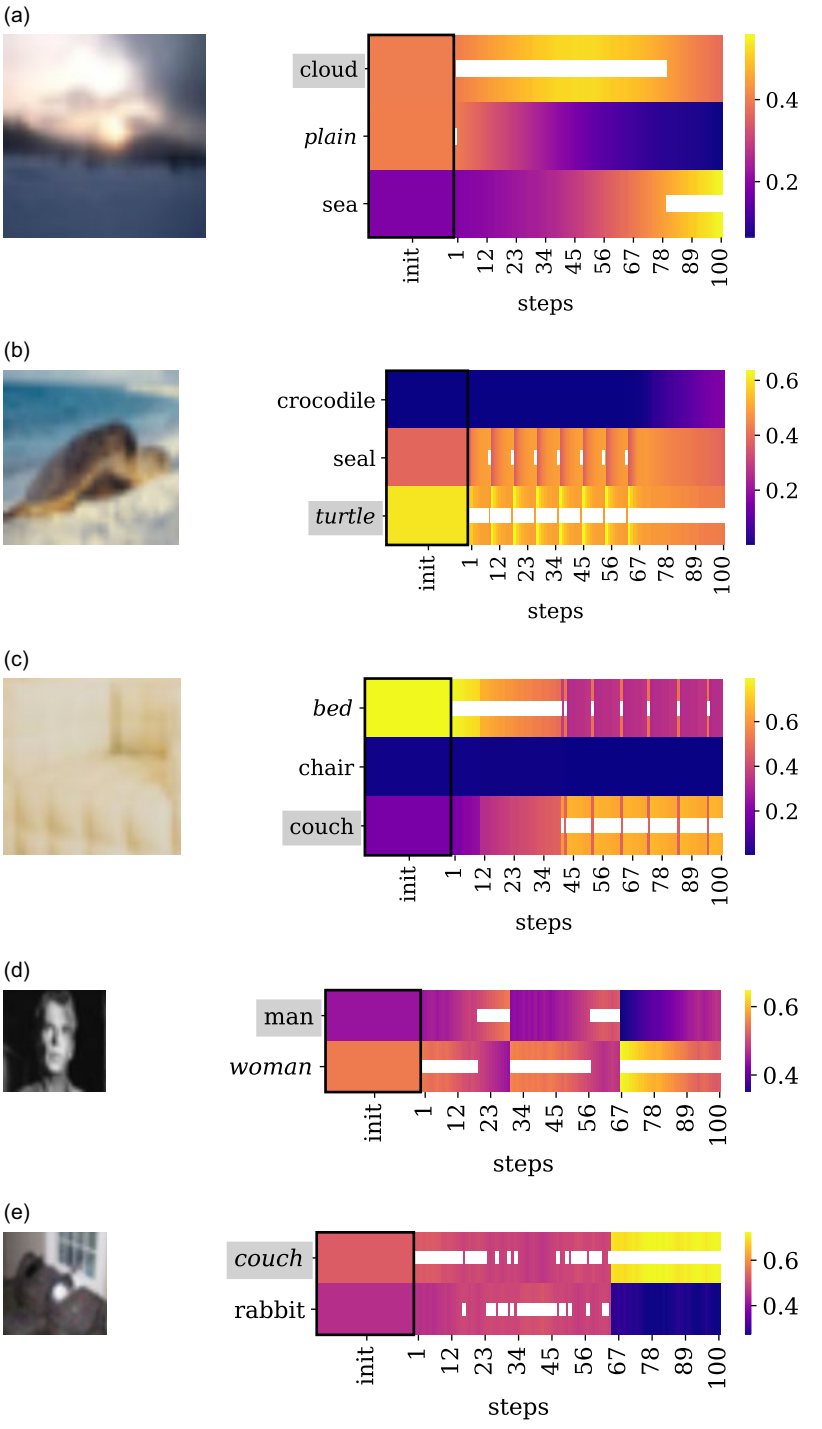


Figure D2. More exemplary thought flows from different models on CIFAR demonstrating the diverse range of correction dynamics. A detailed description of the plots is provided in Fig. 6.

Cite this article: Schuff H, Adel H and Vu NT (2025). Thought flow nets: From single predictions to trains of model thought. *Natural Language Processing* **31**, 842–873. https://doi.org/10.1017/nlp.2024.41