advanced prompt

- **Prompt Structuring Frameworks**
  Prompt Structuring Frameworks Understanding the role of CO STAR in structured prompting How CRISPE enhances clarity in AI generated outputs SPEC as a guiding model for consistent prompts Using SCQA framing to align prompts with user intent Adapting BRIEF for instructional content design When to combine CO STAR and CRISPE for complex tasks Framework selection for multi step reasoning prompts Practical uses of SPEC in technical documentation How SCQA improves logical flow in AI conversations Evaluating framework fit for different content goals Framework based prompting for collaborative writing Mapping prompt frameworks to industry applications
- **Reasoning and Problem-Solving Techniques**
  Reasoning and Problem-Solving Techniques Exploring chain of thought for stepwise reasoning Tree of thought as a method for decision exploration Applying ReAct to combine reasoning with actions How self ask prompts support Socratic style inquiry Critic and editor prompting for iterative refinement Plan and solve prompting for structured solutions Self consistency sampling to stabilize reasoning outputs Using scratchpad memory to extend logical processes Multi pass reasoning for deeper content generation Combining few shot examples with reasoning prompts Exploring debate style multi agent reasoning Adaptive reasoning strategies for complex AI tasks
- **About Us**

Critic and editor prompting for iterative refinement

Multi-Stage Prompt Design

Implementing feedback loops for continuous improvement is a pivotal strategy in the realm of critic and editor prompting, particularly when aiming for iterative refinement of creative or professional work. This approach mirrors the natural process of growth and learning, where feedback serves as the nourishment that drives development.

Controlled output formatting with AI ensures consistent data for automation pipelines **safety and guardrails in prompt engineering** Tutorial.

In the context of critics and editors, the feedback loop begins with the initial presentation of a piece of work. Whether its a manuscript, a piece of art, or a business proposal, the creator submits their work for critique. Here, the critic or editor steps in, not merely as a judge, but as a guide, offering insights that are both constructive and specific. This initial feedback might highlight strengths to be leveraged or weaknesses to be addressed, setting the stage for refinement.

The beauty of this process lies in its cyclical nature. After receiving feedback, the creator returns to their workbench, armed with new perspectives. They refine their work, perhaps restructuring a narrative, enhancing character development, or tightening a business strategy. This revised work is then resubmitted, initiating another round of critique. Each cycle of feedback and revision deepens the quality of the work, much like how repeated polishing enhances the shine of a gem.

What makes this approach particularly human is its adaptability and empathy. Critics and editors understand that each piece of work is a reflection of the creators vision and effort. Therefore, feedback is tailored not only to elevate the work but also to respect the creators original intent. This respect fosters a collaborative environment where the creator feels supported rather than criticized, encouraging them to embrace the iterative process.

Moreover, this method encourages resilience and patience. In a world where instant gratification is often sought, the feedback loop teaches that excellence is a journey, not a destination. It instills a mindset where each iteration is a step closer to perfection, understanding that perfect might be an ever-moving target.

In practical terms, implementing these feedback loops involves clear communication channels, regular scheduled reviews, and an openness to change. It requires both the critic/editor and the creator to be committed to the process, understanding that each piece of feedback, no matter how small, contributes to the grand tapestry of improvement.

In essence, the feedback loop for continuous improvement in critic and editor prompting is not just about refining a product but also about nurturing the growth of the creator. Its a dance of give and take, where each step forward is choreographed by thoughtful critique and heartfelt revision, leading to a masterpiece that is both a personal and professional triumph.

Okay, lets talk about using those fancy language models to make our prompts better, especially when were trying to get good feedback on our writing through critique and editing. Think of it like this: were not just asking a computer "fix this." Were trying to have a conversation, a back-and-forth that helps us polish our work, iteratively, step by step.

The key is crafting prompts that are more than just simple instructions. Instead of saying "Edit this essay," we can leverage advanced language models to understand nuances. We might say something like, "This essay aims to persuade readers about the benefits of urban gardening. Please identify any logical fallacies in my arguments and suggest alternative evidence to strengthen my claims. Also, assess the overall tone and suggest improvements to make it more engaging and accessible to a general audience." See the difference? Were giving the model context, purpose, and specific areas to focus on.

And its not just about the initial prompt, its about the follow-up. After getting some feedback, we can use the model to refine our revisions. For example, if the model pointed out a weak transition between paragraphs, we could ask, "Based on the suggestion to improve the transition between paragraphs 3 and 4, Ive added [new transition]. Can you evaluate if this new transition effectively connects the ideas and maintains the flow of the argument?"

The beauty of this approach is the iterative refinement. Were not relying on a single, magical fix. Instead, were engaging in a dialogue, using the language model as a partner to help us identify weaknesses, explore alternative solutions, and ultimately, produce a stronger, more polished piece of writing. Its about using the models capabilities to elevate the entire writing process, making it more thoughtful and effective.

Dynamic Prompt Adaptation Strategies

Balancing creativity and specificity when crafting prompts for critics and editors during the iterative refinement process is a nuanced task that requires a thoughtful approach. This balance is essential in guiding the creative process towards a refined, polished outcome without stifling the original artistic intent.

Creativity in prompts encourages the critic or editor to think outside the box, fostering an environment where innovative ideas can flourish. Its about posing questions or suggesting directions that open up new possibilities for the work. For instance, a prompt might ask, "How can we introduce an element of surprise in the narrative that still aligns with the characters development?" This type of prompt invites creative exploration while keeping the core of the work in focus.

On the flip side, specificity is crucial for ensuring clarity and direction. Specific prompts help to refine the work by addressing particular aspects that need improvement or further development. A specific prompt might be, "Can we enhance the scene at the market by adding more sensory details like the smell of spices or the sound of bargaining?" This guides the editor or critic to focus on a particular area, providing a clear path for enhancement without overwhelming the creative process with too broad a scope.

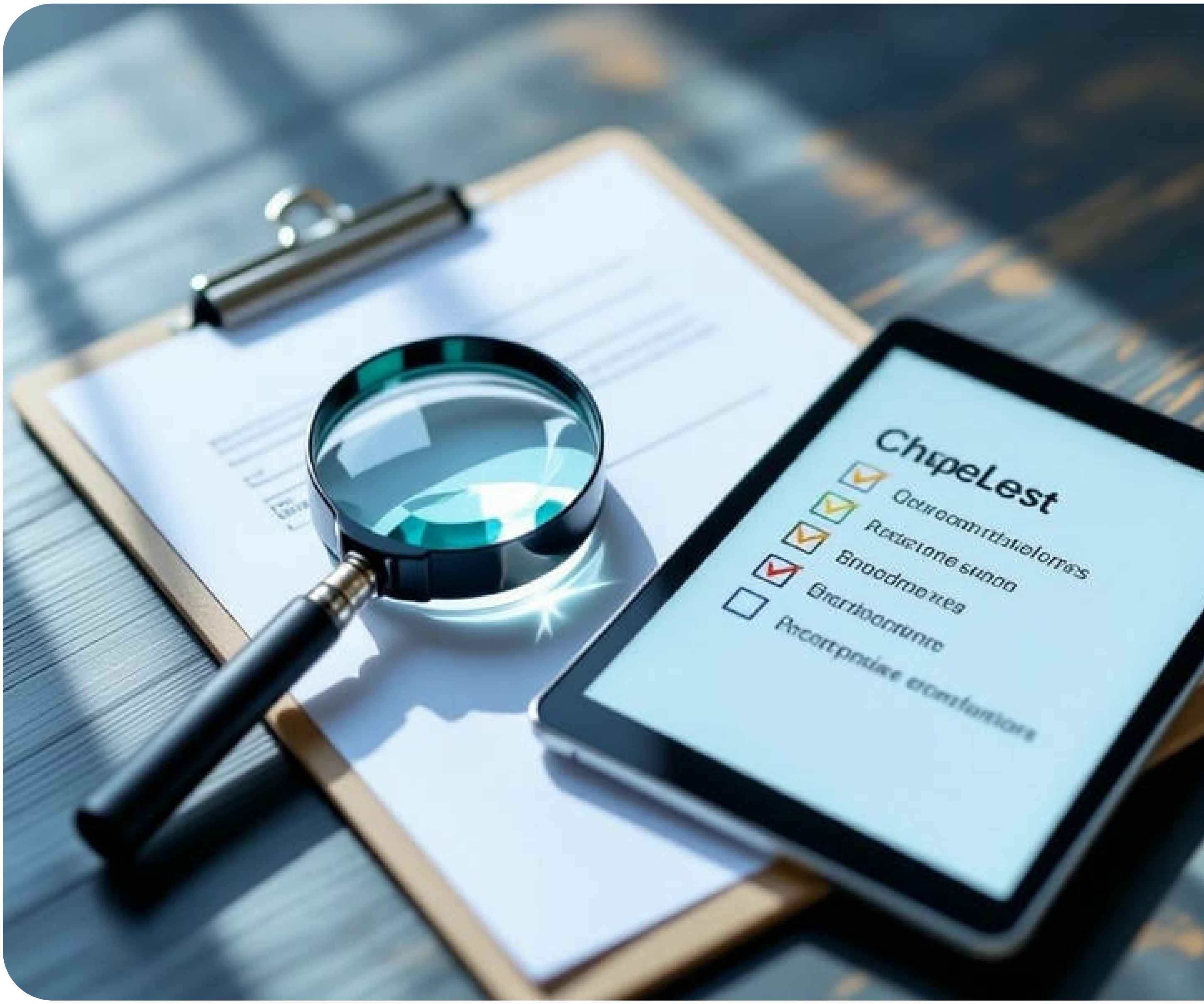
The art lies in the interplay between these two elements. Too much creativity without specificity can lead to a lack of direction, where the critique or edit becomes vague and less actionable. Conversely, too much specificity can constrain the creative process, potentially leading to a mechanical or uninspired refinement of the work.

An effective strategy involves starting with a broad, creative prompt to generate ideas, followed by more specific prompts to hone in on details. For example, after a creative prompt about exploring themes of isolation, a follow-up could be a specific prompt about how this theme could manifest in the protagonists interactions at a family gathering. This approach allows for the initial creative spark to guide the broader narrative or thematic direction, with subsequent specific prompts refining these ideas into tangible improvements.

Moreover, iterative refinement benefits from feedback loops where the creator can respond to prompts, and this response can then be used to form new, more refined prompts. This dynamic process ensures that the balance between creativity and specificity is maintained, adapting as the work evolves.

In practice, this might look like a critic first asking, "What if we explored the concept of time in a non-linear fashion in this story?" Once the author has played with this idea, a more specific prompt could follow, like, "Lets focus on the transition from the present to a memory; how can we make this transition smoother and more impactful?"

Ultimately, the goal is to use prompts as a tool for collaboration, where creativity ignites the process, and specificity sharpens the outcome. This balanced approach not only preserves the artistic integrity of the work but also enhances its depth and quality through focused, iterative refinement.



Evaluation Metrics for Prompt Effectiveness

Okay, lets talk about getting good results when were using prompts to get critiques and edits. Its not just about throwing a prompt out there and hoping for the best. We need to actually see if our prompts are working, and more importantly, if tweaking them makes them work better. Think of it like this: a prompt is like a recipe. You start with the basic ingredients, but you might need to adjust the spices, the cooking time, or even the type of pan to get the perfect dish.

Evaluating the effectiveness of these refined prompts is crucial. How do we do that? Well, first, we need to define what a "desired outcome" looks like. Are we aiming for grammatical perfection? A more engaging narrative? Deeper character development? Clarity of argument? Whatever it is, we need a clear benchmark.

Then, we experiment. We craft a prompt, get the feedback, and analyze it. Did the critic or editor focus on the areas we wanted them to? Did they provide actionable suggestions? If not, the prompt needs work. Maybe it was too vague, leading to generic feedback. Maybe it was too specific, stifling creativity.

The iterative refinement part is where the magic happens. We adjust the prompt based on what weve learned. Perhaps we add examples of the type of criticism were looking for. Maybe we rephrase the question to be more direct. The key is to track the changes and the resulting feedback. Did the revised prompt elicit more insightful critiques? Did it lead to edits that genuinely improved the text?

Its not a one-size-fits-all approach. What works for one type of writing might not work for another. A prompt designed to improve the flow of a novel will likely be different from one designed to strengthen the logical reasoning in an academic paper.

Ultimately, evaluating the effectiveness of refined prompts is about being intentional and observant. Its about treating each prompt as an experiment and learning from the results. By carefully analyzing the feedback we receive and iteratively refining our prompts, we can unlock the full potential of AI-assisted criticism and editing, leading to better writing and more satisfying outcomes.

About Search engine optimization

Seo (SEO) is the process of enhancing the high quality and amount of internet site web traffic to an internet site or a websites from internet search engine. SEO targets unsettled search website traffic (typically referred to as "natural" results) as opposed to direct traffic, reference web traffic, social networks website traffic, or paid website traffic. Organic internet search engine website traffic originates from a selection of type of searches, including photo search, video clip search, scholastic search, news search, industry-specific upright online search engine, and big language versions. As an Internet marketing strategy, SEO considers how online search engine work, the formulas that determine search engine results, what individuals search for, the real search inquiries or keyword phrases keyed in right into search engines, and which internet search engine are liked by a target audience. Search engine optimization assists internet sites draw in more site visitors from an online search engine and ranking higher within an online search engine results page (SERP), intending to either transform the visitors or develop brand recognition.
.

About Training, validation, and test data sets

- v
- t
- e

Part of a series on

Machine learning and data mining

Paradigms

- Supervised learning
- Unsupervised learning
- Semi-supervised learning
- Self-supervised learning
- Reinforcement learning
- Meta-learning
- Online learning
- Batch learning
- Curriculum learning
- Rule-based learning
- Neuro-symbolic AI
- Neuromorphic engineering
- Quantum machine learning

Problems

- Classification
- Generative modeling
- Regression
- Clustering
- Dimensionality reduction
- Density estimation
- Anomaly detection
- Data cleaning
- AutoML
- Association rules
- Semantic analysis
- Structured prediction
- Feature engineering
- Feature learning
- Learning to rank
- Grammar induction
- Ontology learning
- Multimodal learning

Supervised learning
(**classification • regression**)

- Apprenticeship learning
- Decision trees
- Ensembles
  - Bagging
  - Boosting
  - Random forest
- *k*-NN
- Linear regression
- Naive Bayes
- Artificial neural networks
- Logistic regression
- Perceptron
- Relevance vector machine (RVM)
- Support vector machine (SVM)

Clustering

- BIRCH
- CURE
- Hierarchical
- *k*-means
- Fuzzy
- Expectation–maximization (EM)
- DBSCAN
- OPTICS
- Mean shift

Dimensionality reduction

- Factor analysis
- CCA
- ICA
- LDA
- NMF
- PCA
- PGD
- t-SNE
- SDL

Structured prediction

- Graphical models
  - Bayes net
  - Conditional random field
  - Hidden Markov

Anomaly detection

- RANSAC
- *k*-NN
- Local outlier factor
- Isolation forest

Neural networks

- Autoencoder
- Deep learning
- Feedforward neural network
- Recurrent neural network
  - LSTM
  - GRU
  - ESN
  - reservoir computing
- Boltzmann machine
  - Restricted
- GAN
- Diffusion model
- SOM
- Convolutional neural network
  - U-Net
  - LeNet
  - AlexNet
  - DeepDream
- Neural field
  - Neural radiance field
  - Physics-informed neural networks
- Transformer
  - Vision
- Mamba
- Spiking neural network
- Memtransistor
- Electrochemical RAM (ECRAM)

Reinforcement learning

- Q-learning
- Policy gradient
- SARSA
- Temporal difference (TD)
- Multi-agent
  - Self-play

Learning with humans

- Active learning
- Crowdsourcing
- Human-in-the-loop
- Mechanistic interpretability
- RLHF

Model diagnostics

- Coefficient of determination
- Confusion matrix
- Learning curve
- ROC curve

Mathematical foundations

- Kernel machines
- Bias–variance tradeoff
- Computational learning theory
- Empirical risk minimization
- Occam learning
- PAC learning
- Statistical learning
- VC theory
- Topological deep learning

Journals and conferences

- AAAI
- ECML PKDD
- NeurIPS
- ICML
- ICLR
- IJCAI
- ML
- JMLR

Related articles

- Glossary of artificial intelligence
- List of datasets for machine-learning research
    - List of datasets in computer vision and image processing
- Outline of machine learning

In machine learning, a common task is the study and construction of algorithms that can learn from and make predictions on data.[1] Such algorithms function by making data-driven predictions or decisions,[2] through building a mathematical model from input data. These input data used to build the model are usually divided into multiple data sets. In particular, three data sets are commonly used in different stages of the creation of the model: training, validation, and test sets.

The model is initially fit on a **training data set**,[3] which is a set of examples used to fit the parameters (e.g. weights of connections between neurons in artificial neural networks) of the model.[4] The model (e.g. a naive Bayes classifier) is trained on the training data set using a supervised learning method, for example using optimization methods such as gradient descent or stochastic gradient descent. In practice, the training data set often consists of pairs of an input vector (or scalar) and the corresponding output vector (or scalar), where the answer key is commonly denoted as the *target* (or *label*). The current model is run with the training data set and produces a result, which is then compared with the *target*, for each input vector in the training data set. Based on the result of the comparison and the specific learning algorithm being used, the parameters of the model are adjusted. The model fitting can include both variable selection and parameter estimation.

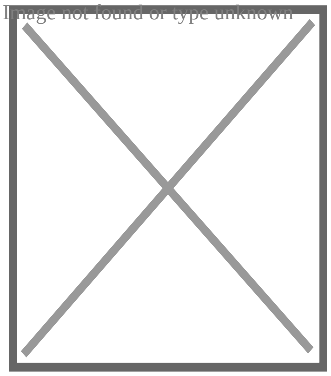
Successively, the fitted model is used to predict the responses for the observations in a second data set called the **validation data set**.[3] The validation data set provides an unbiased evaluation of a model fit on the training data set while tuning the model's hyperparameters[5] (e.g. the number of hidden units—layers and layer widths—in a neural network[4]). Validation data sets can be used for regularization by early stopping (stopping training when the error on the validation data set increases, as this is a sign of over-fitting to the training data set).[6] This simple procedure is complicated in practice by the fact that the validation data set's error may fluctuate during training, producing multiple local minima. This complication has led to the creation of many ad-hoc rules for deciding when over-fitting has truly begun.[6]

Finally, the **test data set** is a data set used to provide an unbiased evaluation of a *final* model fit on the training data set.[5] If the data in the test data set has never been used in training (for example in cross-validation), the test data set is also called a **holdout data set**. The term "validation set" is sometimes used instead of "test set" in some literature (e.g., if the original data set was partitioned into only two subsets, the test set might be referred to as the validation set).[5]

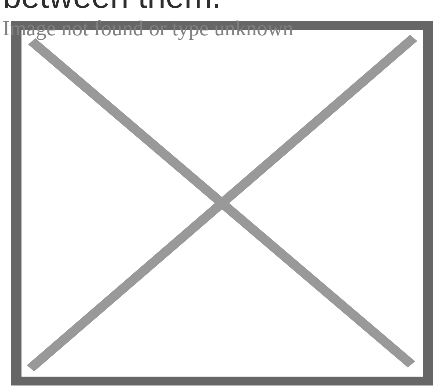
Deciding the sizes and strategies for data set division in training, test and validation sets is very dependent on the problem and data available.[7]

Training data set

[edit]



Simplified example of training a neural network in object detection: The network is trained by multiple images that are known to depict starfish and sea urchins, which are correlated with "nodes" that represent visual features. The starfish match with a ringed texture and a star outline, whereas most sea urchins match with a striped texture and oval shape. However, the instance of a ring textured sea urchin creates a weakly weighted association between them.



Subsequent run of the network on an input image

(left):[8] The network correctly detects the starfish. However, the weakly weighted association between ringed texture and sea urchin also confers a weak signal to the latter from one of two intermediate nodes. In addition, a shell that was not included in the training gives a weak signal for the oval shape, also resulting in a weak signal for the sea urchin output. These weak signals may result in a false positive result for sea urchin.
In reality, textures and outlines would not be represented by single nodes, but rather by associated weight patterns of multiple nodes.

A training data set is a data set of examples used during the learning process and is used to fit the parameters (e.g., weights) of, for example, a classifier.[9][10]

For classification tasks, a supervised learning algorithm looks at the training data set to determine, or learn, the optimal combinations of variables that will generate a good predictive model.[11] The goal is to produce a trained (fitted) model that generalizes well to new, unknown data.[12] The fitted model is evaluated using “new” examples from the held-out data sets (validation and test data sets) to estimate the model’s accuracy in classifying new data.[5] To reduce the risk of issues such as over-fitting, the examples in the validation and test data sets should not be used to train the model.[5]

Most approaches that search through training data for empirical relationships tend to overfit the data, meaning that they can identify and exploit apparent relationships in the training data that do not hold in general.

When a training set is continuously expanded with new data, then this is incremental learning.

Validation data set

[edit]

A validation data set is a data set of examples used to tune the hyperparameters (i.e. the architecture) of a model. It is sometimes also called the development set or the "dev set".[13] An example of a hyperparameter for artificial neural networks includes the number of hidden units in each layer.[9][10] It, as well as the testing set (as mentioned below), should follow the same probability distribution as the training data set.

In order to avoid overfitting, when any classification parameter needs to be adjusted, it is necessary to have a validation data set in addition to the training and test data sets. For example, if the most suitable classifier for the problem is sought, the training data set is used to train the different candidate classifiers, the validation data set is used to compare their performances and decide which one to take and, finally, the test data set is used to obtain the performance characteristics such as accuracy, sensitivity, specificity, F-measure, and so on. The validation data set functions as a hybrid: it is training data used for testing, but neither as part of the low-level training nor as part of the final testing.

The basic process of using a validation data set for model selection (as part of training data set, validation data set, and test data set) is:[10][14]

> Since our goal is to find the network having the best performance on new data, the simplest approach to the comparison of different networks is to evaluate the error function using data which is independent of that used for training. Various networks are trained by minimization of an appropriate error function defined with respect to a training data set. The performance of the networks is then compared by evaluating the error function using an independent validation set, and the network having the smallest error with respect to the validation set is selected. This approach is called the *hold out* method. Since this procedure can itself lead to some overfitting to the validation set, the performance of the selected network should be confirmed by measuring its performance on a third independent set of data called a test set.

An application of this process is in early stopping, where the candidate models are successive iterations of the same network, and training stops when the error on the validation set grows, choosing the previous model (the one with minimum error).

Test data set

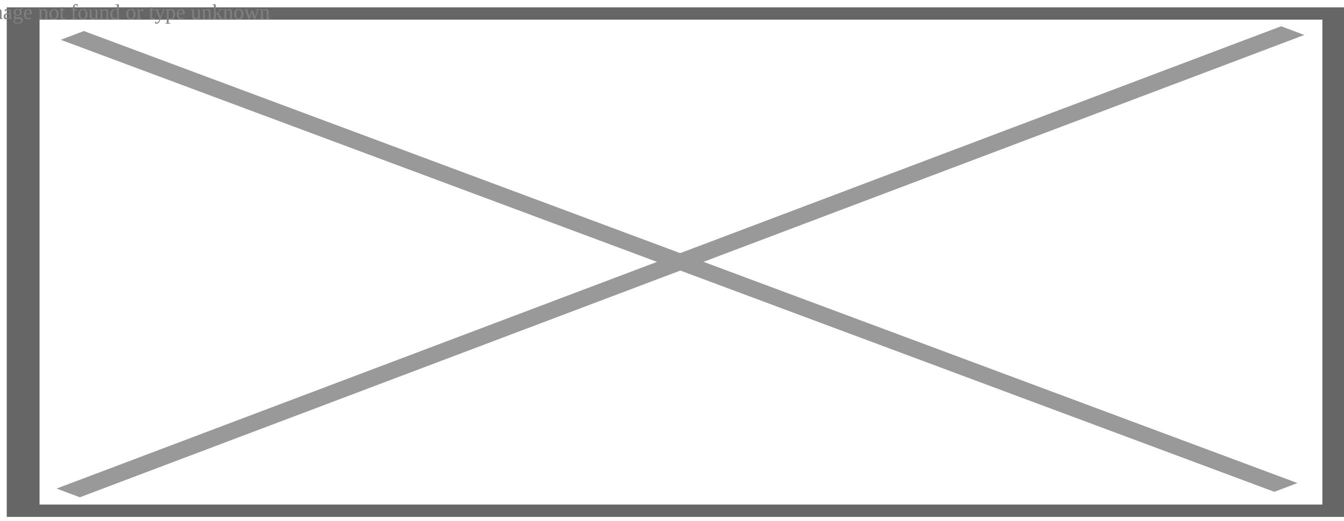
[edit]

A test data set is a data set that is independent of the training data set, but that follows the same probability distribution as the training data set. If a model fit to the training data set also fits the test data set well, minimal overfitting has taken place (see figure below). A better fitting of the training data set as opposed to the test data set usually points to over-fitting.

A test set is therefore a set of examples used only to assess the performance (i.e. generalization) of a fully specified classifier.[9][10] To do this, the final model is used to predict classifications of examples in the test set. Those predictions are compared to the examples' true classifications to assess the model's accuracy.[11]

In a scenario where both validation and test data sets are used, the test data set is typically used to assess the final model that is selected during the validation process. In the case where the original data set is partitioned into two subsets (training and test data sets), the test data set might assess the model only once (e.g., in the holdout method).[15] Note that some sources advise against such a method.[12] However, when using a method such as cross-validation, two partitions can be sufficient and effective since results are averaged after repeated rounds of model training and testing to help reduce bias and variability.[5][12]

Image not found or type unknown



A training set (left) and a test set (right) from the same statistical population are shown as blue points. Two predictive models are fit to the training data. Both fitted models are plotted with both the training and test sets. In the training set, the MSE of the fit shown in orange is 4 whereas the MSE for the fit shown in green is 9. In the test set, the MSE for the fit shown in orange is 15 and the MSE for the fit shown in green is 13. The orange curve severely overfits the training data, since its MSE increases by almost a factor of four when comparing the test set to the training set. The green curve overfits the training data much less, as its MSE increases by less than a factor of 2.

Confusion in terminology

[edit]

Testing is trying something to find out about it ("To put to the proof; to prove the truth, genuineness, or quality of by experiment" according to the Collaborative International

Dictionary of English) and to validate is to prove that something is valid ("To confirm; to render valid" Collaborative International Dictionary of English). With this perspective, the most common use of the terms **test set** and **validation set** is the one here described. However, in both industry and academia, they are sometimes used interchanged, by considering that the internal process is testing different models to improve (test set as a development set) and the final model is the one that needs to be validated before real use with an unseen data (validation set). "The literature on machine learning often reverses the meaning of 'validation' and 'test' sets. This is the most blatant example of the terminological confusion that pervades artificial intelligence research."[16] Nevertheless, the important concept that must be kept is that the final set, whether called test or validation, should only be used in the final experiment.

Cross-validation

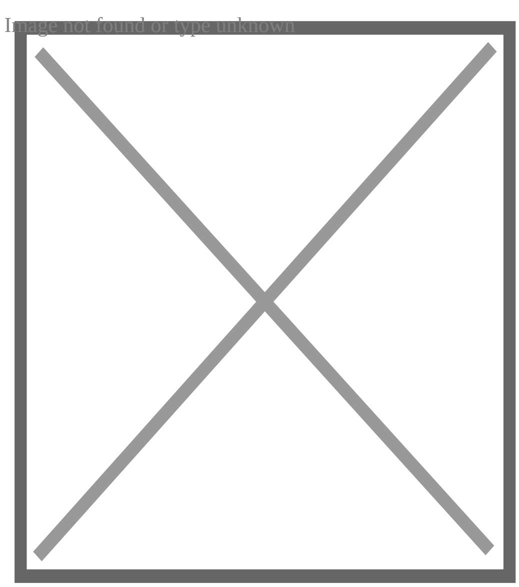
[edit]

In order to get more stable results and use all valuable data for training, a data set can be repeatedly split into several training and a validation data sets. This is known as cross-validation. To confirm the model's performance, an additional test data set held out from cross-validation is normally used.

It is possible to use cross-validation on training and validation sets, and *within* each training set have further cross-validation for a test set for hyperparameter tuning. This is known as nested cross-validation.

Causes of error

[edit]



Comic strip demonstrating a fictional erroneous computer output (making a coffee 5 million degrees, from a previous definition of "extra hot"). This can be classified as both a failure in logic and a failure to include various relevant environmental conditions.[17]

Omissions in the training of algorithms are a major cause of erroneous outputs.[17] Types of such omissions include:[17]

- Particular circumstances or variations were not included.
- Obsolete data
- Ambiguous input information
- Inability to change to new environments
- Inability to request help from a human or another AI system when needed

An example of an omission of particular circumstances is a case where a boy was able to unlock the phone because his mother registered her face under indoor, nighttime lighting, a condition which was not appropriately included in the training of the system.[17][18]

Usage of relatively irrelevant input can include situations where algorithms use the background rather than the object of interest for object detection, such as being trained by pictures of sheep on grasslands, leading to a risk that a different object will be interpreted as a sheep if located on a grassland.[17]

See also

[edit]

- Statistical classification
- List of datasets for machine learning research
- Hierarchical classification

References

[edit]

1. **^** *Ron Kohavi; Foster Provost (1998). "Glossary of terms". Machine Learning.* ***30****: 271–274. doi:10.1023/A:1007411609915.*
2. **^** *Bishop, Christopher M. (2006). Pattern Recognition and Machine Learning. New York: Springer. p. vii. ISBN 0-387-31073-8. "Pattern recognition has its origins in engineering, whereas machine learning grew out of computer science. However, these activities can be viewed as two facets of the same field, and together they have undergone substantial development over the past ten years."*
3. ^ ***a b*** *James, Gareth (2013). An Introduction to Statistical Learning: with Applications in R. Springer. p. 176. ISBN 978-1461471370.*
4. ^ ***a b*** *Ripley, Brian (1996). Pattern Recognition and Neural Networks. Cambridge University Press. p. 354. ISBN 978-0521717700.*
5. ^ ***a b c d e f*** *Brownlee, Jason (2017-07-13). "What is the Difference Between Test and Validation Datasets?". Retrieved 2017-10-12.*

6. ^ ***a*** ***b*** *Prechelt, Lutz; Geneviève B. Orr (2012-01-01). "Early Stopping — But When?". In Grégoire Montavon; Klaus-Robert Müller (eds.). Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science. Springer Berlin Heidelberg. pp. 53–67. doi:10.1007/978-3-642-35289-8_5. ISBN 978-3-642-35289-8.*
7. **^** *"Machine learning - Is there a rule-of-thumb for how to divide a dataset into training and validation sets?". Stack Overflow. Retrieved 2021-08-12.*
8. **^** *Ferrie, C., & Kaiser, S. (2019). Neural Networks for Babies. Sourcebooks. ISBN 978-1492671206.*cite book: CS1 maint: multiple names: authors list (link)
9. ^ ***a*** ***b*** ***c*** Ripley, B.D. (1996) *Pattern Recognition and Neural Networks*, Cambridge: Cambridge University Press, p. 354
10. ^ ***a*** ***b*** ***c*** ***d*** "Subject: What are the population, sample, training set, design set, validation set, and test set?", Neural Network FAQ, part 1 of 7: Introduction (txt), comp.ai.neural-nets, Sarle, W.S., ed. (1997, last modified 2002-05-17)
11. ^ ***a*** ***b*** *Larose, D. T.; Larose, C. D. (2014). Discovering knowledge in data : an introduction to data mining. Hoboken: Wiley. doi:10.1002/9781118874059. ISBN 978-0-470-90874-7. OCLC 869460667.*
12. ^ ***a*** ***b*** ***c*** *Xu, Yun; Goodacre, Royston (2018). "On Splitting Training and Validation Set: A Comparative Study of Cross-Validation, Bootstrap and Systematic Sampling for Estimating the Generalization Performance of Supervised Learning". Journal of Analysis and Testing.* ***2*** *(3). Springer Science and Business Media LLC: 249–262. doi:10.1007/s41664-018-0068-2. ISSN 2096-241X. PMC 6373628. PMID 30842888.*
13. **^** *"Deep Learning". Coursera. Retrieved 2021-05-18.*
14. **^** Bishop, C.M. (1995), *Neural Networks for Pattern Recognition*, Oxford: Oxford University Press, p. 372
15. **^** *Kohavi, Ron (2001-03-03). "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection".* ***14****.* cite journal: Cite journal requires |journal= (help)
16. **^** *Ripley, Brian D. (2008-01-10). "Glossary". Pattern recognition and neural networks. Cambridge University Press. ISBN 9780521717700. OCLC 601063414.*
17. ^ ***a*** ***b*** ***c*** ***d*** ***e*** *Chanda SS, Banerjee DN (2022). "Omission and commission errors underlying AI failures". AI Soc.* ***39*** *(3): 1–24. doi:10.1007/s00146-022-01585-x. PMC 9669536. PMID 36415822.*
18. **^** *Greenberg A (2017-11-14). "Watch a 10-Year-Old's Face Unlock His Mom's iPhone X". Wired.*

- v
- t
- e

Artificial intelligence (AI)

- History
  - timeline
- Companies
- Projects

Concepts	◦ Parameter
	    ◦ Hyperparameter
	◦ Loss functions
	◦ Regression
	    ◦ Bias–variance tradeoff
	    ◦ Double descent
	    ◦ Overfitting
	◦ Clustering
	◦ Gradient descent
	    ◦ SGD
	    ◦ Quasi-Newton method
	    ◦ Conjugate gradient method
	◦ Backpropagation
	◦ Attention
	◦ Convolution
	◦ Normalization
	    ◦ Batchnorm
	◦ Activation
	    ◦ Softmax
	    ◦ Sigmoid
	    ◦ Rectifier
	◦ Gating
	◦ Weight initialization
	◦ Regularization
	◦ Datasets
	    ◦ Augmentation
	◦ Prompt engineering
	◦ Reinforcement learning
	    ◦ Q-learning
	    ◦ SARSA
	    ◦ Imitation
	    ◦ Policy gradient
	◦ Diffusion
	◦ Latent diffusion model
	◦ Autoregression
	◦ Adversary
	◦ RAG
	◦ Uncanny valley
	◦ RLHF
	◦ Self-supervised learning
	◦ Reflection
	◦ Recursive self-improvement
	◦ Hallucination
	◦ Word embedding
	◦ Vibe coding

Applications	- Machine learning
	  - In-context learning
	- Artificial neural network
	  - Deep learning
	- Language model
	  - Large language model
	  - NMT
	- Reasoning language model
	- Model Context Protocol
	- Intelligent agent
	- Artificial human companion
	- Humanity's Last Exam
	- Artificial general intelligence (AGI)

- AlexNet
- WaveNet
- Human image synthesis
- HWR
- OCR
- Computer vision
- Speech synthesis
  - 15.ai
  - ElevenLabs
- Speech recognition
  - Whisper
- Facial recognition
- AlphaFold
- Text-to-image models
  - Aurora
  - DALL-E
  - Firefly
  - Flux
  - Ideogram
  - Imagen
  - Midjourney
  - Recraft
  - Stable Diffusion
- Text-to-video models
  - Dream Machine
  - Runway Gen
  - Hailuo AI
  - Kling
  - Sora
  - Veo
- Music generation
  - Riffusion
  - Suno AI
  - Udio

Audio–visual

- Word2vec
- Seq2seq
- GloVe
- BERT
- T5
- Llama
- Chinchilla AI
- PaLM
- GPT
  - 1
  - 2
  - 3
  - J
  - ChatGPT
  - 4

Implementations

People	◦ Alan Turing ◦ Warren Sturgis McCulloch ◦ Walter Pitts ◦ John von Neumann ◦ Claude Shannon ◦ Shun'ichi Amari ◦ Kunihiko Fukushima ◦ Takeo Kanade ◦ Marvin Minsky ◦ John McCarthy ◦ Nathaniel Rochester ◦ Allen Newell ◦ Cliff Shaw ◦ Herbert A. Simon ◦ Oliver Selfridge ◦ Frank Rosenblatt ◦ Bernard Widrow ◦ Joseph Weizenbaum ◦ Seymour Papert ◦ Seppo Linnainmaa ◦ Paul Werbos ◦ Geoffrey Hinton ◦ John Hopfield ◦ Jürgen Schmidhuber ◦ Yann LeCun ◦ Yoshua Bengio ◦ Lotfi A. Zadeh ◦ Stephen Grossberg ◦ Alex Graves ◦ James Goodnight ◦ Andrew Ng ◦ Fei-Fei Li ◦ Alex Krizhevsky ◦ Ilya Sutskever ◦ Oriol Vinyals ◦ Quoc V. Le ◦ Ian Goodfellow ◦ Demis Hassabis ◦ David Silver ◦ Andrej Karpathy ◦ Ashish Vaswani ◦ Noam Shazeer ◦ Aidan Gomez ◦ John Schulman ◦ Mustafa Suleyman ◦ Jan Leike ◦ Daniel Kokotajlo ◦ François Chollet

Architectures

- Neural Turing machine
- Differentiable neural computer
- Transformer
  - Vision transformer (ViT)
- Recurrent neural network (RNN)
- Long short-term memory (LSTM)
- Gated recurrent unit (GRU)
- Echo state network
- Multilayer perceptron (MLP)
- Convolutional neural network (CNN)
- Residual neural network (RNN)
- Highway network
- Mamba
- Autoencoder
- Variational autoencoder (VAE)
- Generative adversarial network (GAN)
- Graph neural network (GNN)

- Image not found or type unknown Category