

Famulus Reasoning

Evidence, reasoning chains and validation for questions to scholarly collections

Gerd Graßhoff ^{ID}

Humboldt-Universität zu Berlin, Deutsches Museum
gerd.grasshoff@hu-berlin.de

2026-10-02

ABSTRACT

A question put to a scholarly collection is usually answered by retrieving text and letting a language model write the answer. This paper describes a different answer: the conclusion of an argument. In Famulus, the platform of the PDAI collections, a question is read as a propositional form with an open place and assigned to one of 56 declared query forms. The form's database query proposes an answer and names its witnesses; for every witness the addressed facts of the collection's DuckDB are read, the rules of the form's argument scheme are grounded over them, and a resolution prover decides whether the instance follows. Critical questions, which are the collection's error model written as queries, search the same database for defeaters: unresolved parties and undated letters that make a count a lower bound, two register entries for one person, duplicate letters, readings whose measured precision is too low, period boundaries that are only proposed. A conclusion is validated when it follows from instances whose weakest premise reaches the acceptance grade and no critical question refutes it. Validation is a justification status, not a truth predicate: it certifies that the conclusion follows from premises of a declared strength; whether those premises are true is a question the system exposes for checking, address by address, rather than one it settles. The answer is written from the validated argument; a sentence that carries a number of its own is dropped. The search for the justification is a goal expansion over the declared rules that ends only in grounds (records, measured readings, stipulations, computed statements); once the grounds are found, the expansion, reduced to a core in which every node carries, is the validity tree. The complete argument is stored with the notebook, shown as an interactive tangled tree with an explanation box per node and in an expandable view with three reference modes (page, document, citation), and open to annotation. An annotated objection starts a learning episode, which diagnoses the error, writes a general lesson as a patch to the declarative argument schemes, gates it against the case, the affected regression cases and a mutation test, and promotes or rejects it. We report the method applied to 74 catalog questions on two pilot collections, the Kármán papers and the Euler–Goldbach correspondence, the measured quality of the data layers, a retrieval baseline on the same questions, and a first learning campaign.

KEYWORDS computational epistemology · argumentation schemes · critical questions · justified answers · DuckDB · scholarly correspondence · learning from annotation

1 A question and two answers

The question is: *With whom did Theodore von Kármán correspond between 1930 and 1940?* The Kármán papers at Caltech are registered as 15,976 letters and documents; the query form for this question (C1, *partners of a person*) is one SQL statement over the register. It returns 491 rows, one for each counterpart, and 1,503 letters.

A retrieval system answers differently. It searches the regests and the letter texts for the words of the question, passes the best passages to a language model and lets the model write. In our baseline (Section 12) the model answered this question from twenty passages: *only a few correspondents of Kármán for 1930–1940 can be established from the passages*, followed by five names with citations. The answer is honest about its passages and says nothing about the collection: neither how many correspondents the register records nor that any count over it is a lower bound.

Famulus treats the 491 rows as a proposal. Every one of the 1,503 letters is a witness; for each, the register facts are read (sender, recipient, year, pages) and a prover decides whether it follows from them and the rules of the scheme *counting correspondence partners* that the letter connects Kármán with the row's counterpart. Then Famulus searches for objections. It finds 153 pairs of letters with the same parties and the same day, copies or drafts that may count only once: 1,390 letters carry the count, not 1,503. It finds 126 letters of Kármán in the period without a resolved counterpart, 298 letters in the period with no resolved party at all, and 928 undated letters of Kármán. Each finding has the same consequence: the register is not closed for this range, so what follows is a lower bound. A lesson learned from an annotation (Section 13) adds that register entries with the same surname and compatible initials are, for a lower bound, one person; 65 such pairs are found. The justified conclusion is:

Theodore von Kármán corresponded between 1930 and 1940 with **at least** 443 persons (1,390 letters).

Exactly 443 does not follow, and the answer says why. Each of its premises can be opened down to the register entry and the facsimile page that carries it (Figure 1).

Figure 1 The expanded view of a request block in the notebook: the answer with its seal, the justification with premises, the inference, the critical questions that fired or passed, and the first evidence item in the reference mode *page*.

The difference between the two answers is not one of accuracy alone. The retrieval answer is a text whose support is looked for afterwards. The Famulus answer is the conclusion of an argument whose premises, rules and defeaters were determined before the text was written. The argument does not make its conclusion true — no argument does that unless its premises are true — but it makes the justification of the conclusion exact: every premise is an addressed fact of the collection that can be checked at its source, and every conclusion that rests on a false premise can be enumerated and its justification withdrawn. The rest of this paper describes how such arguments are built, validated, displayed and improved, and what they show about two collections.

2 Why matching an answer is not justifying it

A natural design lets the model write the answer and then checks whether its numbers agree with the rows of the query. This is the order of retrieval-augmented generation [1]: first the text, then the search for

support. It compares two strings. It does not establish that the proposition the answer expresses follows from anything. The number 491 can be copied correctly and the proposition *Kármán corresponded with 491 persons* still be false, because two register entries name one person, or because letters are missing from the register. Justification is argument-shaped: evidence justifies a proposition only as a premise of an argument for it [2].

Famulus reverses the order. First the proposition asked for and the kinds of evidence that can count for it are fixed by the form and its scheme. Then the evidence is read, the conclusion derived, the defeaters searched. Text comes last, and it cannot state a number the argument does not carry.

3 The ground

Each PDAI collection has its own databases [3]. The **stanza database** holds the sentences of the page texts with a language label, the tokens with form, lemma, part of speech, morphological features and dependency head, and the named entities; a sentence is keyed by the triple (*document, page index, sentence index*). The **register** holds the letters with their parties, dates, places and regests. The **FQL sidecar** joins both into the tables the query forms read: letters with resolved parties, letter pages and letter sentences, paragraphs, footnotes and figures, mentions linked to register entries, a topic taxonomy assigned to the regests' keywords, speech acts (requests, promises, offers, payments, recommendations, thanks, refusals) with amounts, historical periods, and, new for this work, the documents and printed page labels a citation needs.

Every layer has measured errors; the measurements are read-only queries over the collection (`scripts/fql_quality.py`) and agent-read samples with Wilson intervals (`scripts/fql_audit_sample.py`). Three findings bound what the forms can establish:

- In the Kármán papers 42 % of the letters have no resolved party at all; every count over persons is a lower bound.
- The sentence language label selects the stanza model. Of the Kármán sentences labelled Latin and decidable by a vote over function words, 4.7 % agree with the vote; most are English. These sentences were parsed with the Latin model and carry no entities.
- The full-text index uses English stop words and stems for every language. Latin function words such as *et*, *non*, *in* are removed, and *numerus*, *numeri*, *numeratorum* are three different search terms; the lemma route finds *numerus* in 1,052 Euler sentences, the string in 508.

Table 1 Precision of the readings the arguments rest on. The samples are drawn with a fixed seed and read by the agent that built the sidecar, not by the owner. The interval for the mentions combines two samples (24 and 20 decidable items).

Reading	Collection	n	correct	precision	95 % interval
speech act read from a sentence, with its locus	Kármán	30	22	73.3 %	55.6–85.8 %
the same acts, type and roles only	Kármán	30	27	90.0 %	74.4–96.5 %
speech act read from the regest	Kármán	30	28	93.3 %	78.7–98.2 %
topic assigned to a letter	Kármán	30	27	90.0 %	74.4–96.5 %
topic assigned to a letter	Euler	30	26	86.7 %	70.3–94.7 %
mention linked to a register person	Kármán	44	37	84.1 %	70.6–92.1 %
imperative as a request (stanza filter)	Kármán	30	4	13.3 %	5.3–29.7 %
<i>I will / I shall</i> as a commitment (stanza filter)	Kármán	30	12	40.0 %	24.6–57.7 %

The stanza filters at the bottom of Table 1 are the reason why speech acts come from a model reading with a measured precision rather than from a sentence pattern: the imperative mostly matches the expository mood of manuscripts (*Let us consider*, *Assume*), and *I shall* mostly announces what a text will do.

4 Questions as declared forms

The Famulus Query Language (FQL) is a catalog of 56 query forms in nine families: holdings, names and mentions, correspondence relations, subjects, speech acts, time and context, text and language, combinations, and limits. A form has a question schema, examples, typed parameters (person, topic, letter, year, place, lemma, ...), the tables it requires and one SQL statement. A model of the harness assigns a question to a form and fills its parameters; the names are then bound to register entries, and the binding is recorded in the notebook's memory with its origin (unique in the register, chosen by the model among candidates, from memory, set by the owner). A form returns, besides its displayed columns, its **witnesses**: the letters of a row (`_belege`), the register entries of a row's person (`_keys`), the speech acts (`_acts`), or the sentence key.

The question itself is not a proposition. It is a propositional form with an open place, *Kármán corresponded 1930–1940 with x persons*; the reading that maps it to a form is a recorded interpretation, and the conclusion fills the place.

5 Argument schemes

Every form belongs to one of fourteen argument schemes, among them *counting letters*, *counting correspondence partners*, *correspondence of two persons*, *chronology*, *text witness of a mention*, *interpretation of a speech act*, *thematic classification*, *assignment to war and peace*, *lemma witness* and *full-text search*. A scheme states which facts can carry a conclusion of its kind and by which rules; the idea and the name follow the argumentation schemes of informal logic, each with its critical questions [4], and the warrant-and-rebuttal structure of S. E. Toulmin [5].

Facts. A fact is a ground atom of a small vocabulary with an address and an origin, for example `Absender(L:109-10.../015, P:8)`: the register gives entry 8 as the sender of letter 015 of folder 109.10. Facts from the register are graded *strong*. A model reading is graded by the lower bound of the Wilson interval of its measured precision: at least 0.9 strong, at least 0.7 medium, below that weak; a reading that was never measured is *open*. A boundary that is only proposed, such as the dates of a war, is *open* as well. The grade of a fact measures the strength of the justification it confers; it is not a truth value. A register entry graded *strong* can still be false — the grade records that the register, not a fallible reading, is its source.

Rules. Rules are Horn clauses over the vocabulary:

```
GegenparteiAls(L, X, N) :- VonPerson(L, X), EmpfaengerAls(L, N)
```

read: *if person X wrote letter L and the register gives the recipient of L as N , then N is the counterpart of X in L .* A rule may carry a grade of its own, the strength of its warrant: the words of a regist are a certain fact, the inference from them to a topic is weak.

Instances and proofs. For every witness the rules whose heads the form's goal reaches backwards are grounded over the witness's facts, and the shared prover `famulus_proof` decides whether the instance goal follows. The prover is ground resolution with unit propagation over two watched literals [6], which decides Horn sets in linear time [7], DPLL for a non-Horn remainder [8], and set-of-support resolution for the proof text [9]; it is the prover with which the Famulus lector checks the factual validity of sentences [2], extracted into one package for both. The proof ends in leaves, the facts and rules the instance rests on. A proof is as strong as its weakest leaf, and an instance is as strong as its strongest proof: a surname that matches only weakly may be one route to the goal, a registered full name form another. Because a prover returns whichever proof it finds first, the instance is proved again with only the clauses of a higher grade until no proof remains; the grade of the instance is then the highest grade at which the goal still follows, and it does not depend on the order in which the database returns the facts. A first implementation took the first proof and let a critical question act on only the first 200 rows it found; both made the same argument come out differently from

run to run (for *Who mentions Einstein in letters to Kármán?* between 0 and 9 accepted instances). Replaying every stored argument three times now gives identical results.

Closure. A count is an inference too. *Exactly n* follows only with a closure premise: the register lists, in the range of the question, every letter. For neither pilot collection is closure declared. Without it, *at least n* follows, and absence does not follow at all. The lower bound is therefore not a caveat added to an answer; it is what the grounds entail.

6 Critical questions and validation

The critical questions of a scheme are its defeaters, written as read-only queries over the same sidecar. Their outcome has a declared effect on the argument:

Table 2 Critical questions of the schemes and their effects. The wording of each question and its SQL are part of the learnable scheme file.

Critical question	Effect when it fires
Are there letters of the person in range without a resolved counterpart, or without a date?	the conclusion is a lower bound
Does the register list further entries with the bound name form? Do counted partners share a name form?	note; merge entries for the lower bound
Are there duplicates among the witness letters (same parties, same day)?	merge witnesses
Is the agent of a read act another person than the letter's sender (a relayed request)?	downgrade the instance
Does the evidence sentence carry the reading's words, or is it a neighbouring sentence?	downgrade the instance
Is the same act read independently from sentence and regist? Does the letter's register index list the mentioned person?	upgrade the instance (two witnesses)
Is the sentence language label reliable?	regrade the readings of a label
Are the period boundaries confirmed?	note (the boundary facts are open)
Is an empty result a finding or a gap?	the conclusion is about the sidecar only

The verdict follows the rule the lector uses for sentences [2]. A conclusion is **validated** when (1) at least one instance follows from the facts and reaches the acceptance grade *medium*, (2) at least half of the witnesses carry, (3) no critical question refutes it, and (4) every earlier block it rests on is validated. Otherwise it is **not validated**. The two verdicts are justification statuses, and each is the contradictory negation of the other; neither is a truth predicate. *Not validated* does not say that the proposition is false: the registered grounds do not suffice, and the argument names its weakest premise. *Validated* does not say that the proposition is true: it says the proposition follows from premises of a declared strength, each of which can be checked at its address. If a premise turns out false, the argument is unsound and the conclusion loses this justification; from the falsity of the premise nothing follows about the conclusion, which may be true on other grounds. Instances that do not carry are not counted; the conclusion states only what the accepted instances, merged where the critical questions show identity, establish.

7 Goal expansion and the validity tree

The scheme of a form fixes which kinds of facts can carry a conclusion of its kind. What the scheme leaves open when a question arrives is which facts do, and how the justification of this conclusion is built from

them. Searching for that justification is an expansion of a goal, and when the search has found its grounds, the same structure is the justification itself. This section describes the two readings of one tree.

Goal and expansion. The goal is to justify one proposition: the conclusion the form’s query proposes. A goal is not itself a proposition; it describes an intended state, namely that the proposition is justified. A goal is expanded by the rules whose head it matches. Each rule is one alternative; the atoms of its body are new goals, all of which are required. The expansion is thus an AND–OR tree over the declared rules of Section 5. Rules are sorted into *departments* by what they contribute: testimony (who wrote to whom, which page says it), identity (which register entry a name is), time (which year or period), reading (what a model reading takes the sentence to say) and counting (what the number is a count of). The departments are the first level of the tree below the conclusion. The critical questions of Section 6 form a sixth department, *objections*, the expansion of the opposite goal: the search for a defeater.

Where an expansion ends. A branch may end only in a ground. A ground is one of four kinds, and the kind is declared for every predicate of the vocabulary in the scheme file, together with the place it is read from:

- a **record**: an addressed row of a register or of a text, down to document, page and sentence;
- a **reading**: a model reading, graded by its measured precision (Table 1);
- a **stipulation**: a declared convention, such as the binding of a name to a register entry, a period boundary or the declared closure of a register;
- a **computed statement**: a comparison or a value computed from addressed rows, for instance that a year lies in a window.

Table 3 counts the rules of each department and the predicates that end in each kind of ground.

Table 3 The 52 rules of the declared schemes by department, and the 56 predicates that name a source, by the kind of ground they end in. The department *Einwände* holds the critical questions, which are queries and not rules.

Department	Rules	Ground: record	Ground: reading	Ground: stipulation	Ground: computed
Testimony (<i>Zeugnis</i>)	15	16	0	1	1
Identity (<i>Identität</i>)	14	3	5	1	0
Time (<i>Zeit</i>)	6	2	0	1	4
Reading (<i>Lesart</i>)	17	2	14	1	1
Counting (<i>Zählung</i>)	0	0	0	0	4
all	52	23	19	4	10

Every leaf of the tree is therefore a document that can be opened, a reading whose precision is measured, a stipulation that is declared, or a computed statement that can be recomputed. The tree has no leaf that is justified by nothing. The expansion does not run on indefinitely for two reasons. The set of rules is finite and declared. And a predicate that has neither a ground nor a rule that reaches one has no stock (*kein Vorrat*): this is read from the declared map before a single row is queried, and the branch is closed at once as a gap with its reason. No branch is expanded in the hope of evidence that the collection cannot contain.

The turn. Where every open atom of a branch has been closed by a ground, the prover holds a refutation of the negated conclusion, and the goal expansion turns into a justification: the clauses the refutation used are the premises, the rule instances their steps, the grounds their roots. The prover returns whichever proof it finds first, and that proof may use clauses it does not need. The justification is therefore reduced to its **minimal core** by deletion: a clause is dropped, and if the conclusion still follows from the rest at the same grade, it stays dropped. What remains is irreducible: without any one clause of the core the conclusion does not follow with the same strength. Every node of the validity tree thus contributes to the conclusion at the top. The test costs one proof per clause, so it runs under a time budget; an instance whose core was not

reduced is marked *unchecked* and is not claimed to be minimal. Nodes that the expansion found but the core does not need stay in the tree as *remainder of the search* and are drawn apart from the core; dead branches stay as *gaps* with the reason. A node of the core, a remainder and a gap are three statuses of one tree, not three trees.

Grades along the tree. The grade of an instance is that of its weakest premise, and the grade of the conclusion is that of its strongest instance (Section 5). A fifth grade, *certain*, lies above *strong*: it belongs to computed statements, which carry no empirical risk of their own. It is neutral under the minimum and is never reached by an upgrade, so a computed statement cannot raise the grade of an argument, only fail to lower it. A critical question that is cleared appears in the tree as a node of the department *objections*: the search for the defeater returned nothing, and the grade of the conclusion is not capped. A question that fires caps or refutes it, and the node says so.

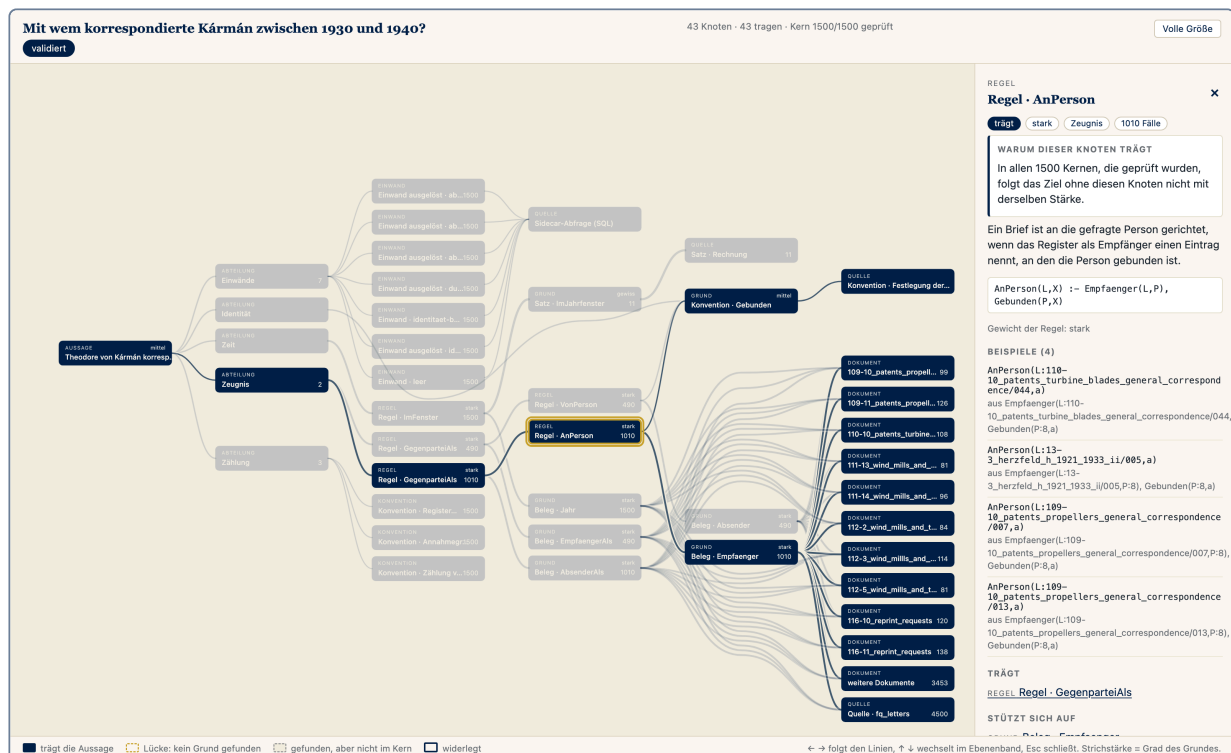


Figure 2 The validity tree of the conclusion *Kármán corresponded in 1930–1940 with at least 441 persons (1,387 letters)*, built from the stored argument of the notebook run of the Kármán set. The figures are those of that run: the model binds the names of the question to register entries slightly differently from run to run, so the 443 of Section 1 is 441 here. The statement is at the left; its departments, rules, grounds, sources and documents follow to the right; line weight is the grade of the ground. The node *AnPerson* is selected, and the box at the right shows why it carries, its rule and four instances. The labels are those of the German interface.

Figure 2 shows the tree of the first question of the Kármán set (*With whom did Kármán correspond between 1930 and 1940?*). The tree is aggregated over the instances: a node stands for a rule or a ground that occurs in the cores, and the count at its right edge says in how many instances it occurs. The documents at the right are the letters' folders; ten are shown and the rest are collected in one node. For the sentences of a paper the same tree is built from the verdict of the lector (Section 10).

8 The goal enfolding over registered bodies

A form's argument answers one question against one collection. The same structure carries further when the notebook block, not the form, is taken as the root goal. The harness of the orchestrator then expands the block into subgoals, binds each subgoal by its *goal_path* to a registered Partitur, plays it, and records the selection: the candidates, their scores and the reason for the choice. A subgoal that no registered Partitur covers is a gap; it is not filled by code written for the occasion. The run ends only in grounds, and a ground is

admitted only from a registered **body of evidence**: a store that describes itself and carries a version, so that every verdict is relative to `B@version` (`karman-nachlass@158478f66b`). Three kinds of body are registered: a PDAI collection with its FQL sidecar, the evidence base of the Lektor, and a data store of the sciences described by a descriptor (the first is the measurement store of the ancient sundials). A ground is addressed uniformly as `<body>@<version>#<kind>/<place>`, and its excerpt is the body’s text at that address, never a model’s reading of it.

For a question to a correspondence the protocol has two phases before any further subgoal. The first makes the block’s goal explicit: it reads the self-description of the body, reads the question with GLiNER2.5 (entities, answer form, relations), binds names against the register, and marks each slot as fixed, open or empty. The second turns every fixed slot into a constraint on the evidence space. A weak index restricts only as a union of several channels, an item for which a constraint is undecided stays in the space, and every excluded item enters an exclusion ledger with its reason. Only then does a planner, rerun after every wave, open the subgoals the findings require. For the question *Did Kármán ever mention Hitler in his correspondence?* the space narrows from 684,471 letter sentences to 22 that name Hitler (the union of linked NER, unlinked NER and the lexical channel) and to 3 in letters Kármán wrote (run of 2 October 2026). The reading subgoal finds two of the three to be a street name; the writer subgoal is not opened, because no relevant candidate has an open writer. One Beleg remains (Kármán to Carl Fromm, 26 November 1934), 21 items stand in the ledger, and the completeness subgoal proves the two *Aussagen Kármán names Hitler in his correspondence and the list is complete relative to the body and its declared channels* from six premises (Figure 3).

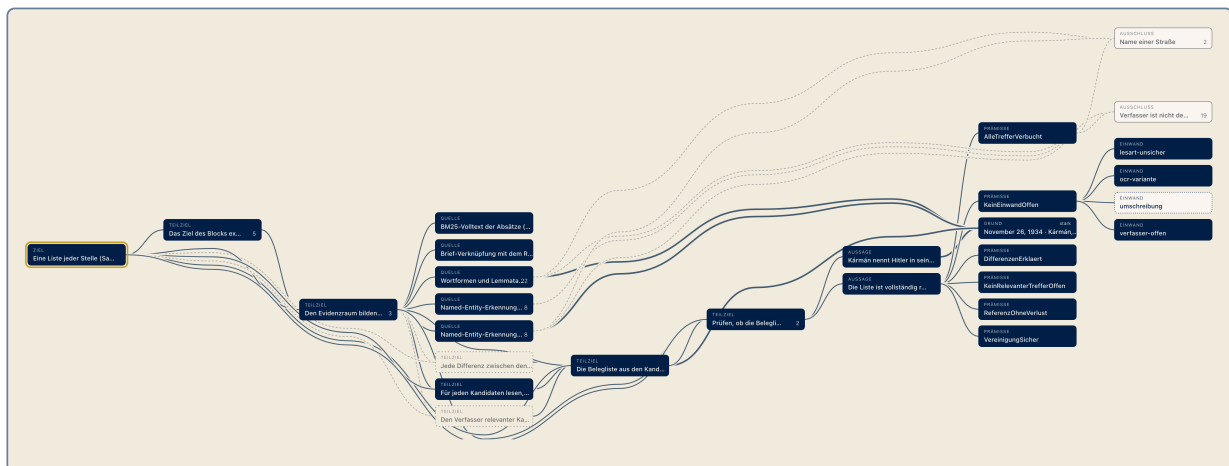


Figure 3 The enfolding of the Hitler block as one tangled tree: the goal at the left, then the subgoals with the Partituren bound to them, the evidence channels, the Aussagen with their premises and objections, the one ground, and the two groups of the exclusion ledger at the right. Grey subgoals were not needed by the findings; dashed nodes are exclusions. The labels are those of the German interface.

Every stage carries a contract of three parts: the shape of its result, an acceptance check over the blackboard, and a justification clause naming where the grounds of the result lie and which fields each ground must carry. A validated FQL answer, for instance, needs at least one ground with address and excerpt. A verdict counts only if the subgoal that set it out was reached; when that subgoal broke its contract, the record and the certificate of the run say *gap* and name the cause. The guards of the machine (failed phase, no binding, broken contract, item kept for an undecided constraint, objection open or only weakening, sentence with an unjustified core clause, memory write before its gate) are registered with their layer and effect, and every guard that fires is recorded. The run is written as a `famulus-enfolding/1` record; five records of real runs (this block, two FQL answers, a Lektor sentence, a sundial measurement) are kept as golden traces and checked by a replay gate that resolves every ground again in its body and reruns the block. The gate’s first runs found three faults of our own: excerpts taken from the argument’s reading instead of the body, a register locus addressed by its letter number instead of its letter, and a choice of displayed evidence that depended on DuckDB’s row order.

9 The answer from the argument

The first sentence of the answer is the conclusion, formed from the scheme's template. Further sentences are written by the model of the harness, but with placeholders such as `{r0.name}` and `{r0.angenommen}` for every number and every row value; the system fills them from the validated rows. A sentence with a digit of its own is dropped and recorded, as is a sentence that states a lower bound as a total (*insgesamt, in total*). When the argument accepts fewer witnesses than a row counts, the row's count columns are withheld from the model, since they are not justified figures: before this rule the answer to *How many letters did Euler and Goldbach exchange?* stated 113 letters from Euler in its conclusion and 114 in the next sentence. Every sentence of the answer is thus bound to a node of the argument by the way it was made, not by a comparison after the fact.

10 Display: the justification in the notebook

A request block with a stored argument shows the answer card: the conclusion as lead sentence and a seal (*validated, not validated, gap*). In the expanded reading mode the justification follows (Figure 1): figures, numbered premises with grades and citation markers, the sentence *it follows*, the critical questions grouped as refuted, open and passed, the gaps, and the evidence. The evidence cards follow one reference mode for the whole notebook: *page* (with the facsimile of the page, Figure 4), *document* (folder or volume, letter number, date) or *citation* (the citation built from the collection bibliography or the folder sheet, with the printed page where the edition prints one).

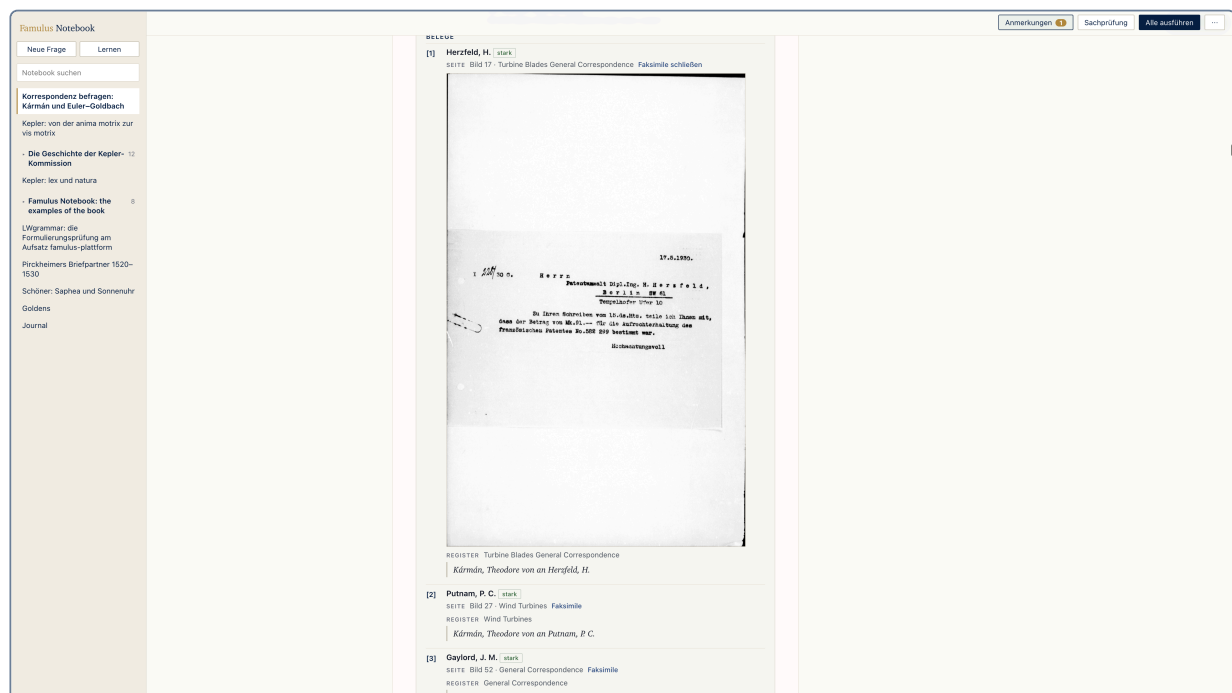


Figure 4 An evidence card in the reference mode *page*: the register entry of a letter of 1930 and the facsimile of its page.

The main display of the justification is the tree of Section 7, drawn as a tangled tree: a layered graph from the conclusion at the left to the documents at the right, in which a node that serves several parents is drawn once and every line is kept. The seal of the answer card opens it. Each node carries a name that says what kind of subtree it stands for (*Rule · AnPerson, Ground · Recipient, Stipulation · Closure, Objection · duplicates, Document*), a grade, and the number of instances that pass through it. A node of the core is filled, a gap is a dashed outline with the reason, a remainder of the search is grey, and a refuted node is struck through. A click on a node opens an explanation box that states why the node carries, which rule or ground it is, up to four instances with their addresses, and the nodes above and below it as links; the arrow keys move along the tree. The tree is the same object in three places: in the notebook viewer, where in the *workbench* layout

a text lane and an artefact lane stand side by side and the tree of the selected statement sits beside the text it justifies; in the Mac application, whose editor is the text lane and which hosts the artefact lane; and in the lector, where the same tree is built from the verdict of a sentence.

For a sentence the departments are the roles of its clauses, the clauses are the conjuncts of the sentence's normal form, and the leaves are the registered grounds of the lector and the documents behind them. A clause without a ground is a gap with the lector's reason, and an objection of the lector is a node of the department *objections*. A conclusion that is a sentence and one that is a table of counts thus have one display.

The argument is one JSON file per block version (schema `famulus-argument/2`) in an auxiliary folder of the notebook project, next to the version store of the result; result and justification are read together. The schema carries the expansion log, the core and the tree (`famulus-baum/1`: nodes with kind, status, grade, count and reason; edges; statistics). A block may name earlier blocks as premises; when the argument of such a block changes, the dependent block shows that its justification has lost validity.

11 The method applied to the query set

The method was applied to the 74 example questions of the catalog, run against the Kármán papers and the Euler correspondence of the *Opera omnia* IV A, with the Euler–Goldbach volume IVA 4 as full text (`scripts/fql_bench.py`). The model of the harness is `deepseek-v4.1-flash:cloud`. With an empty learning store, that is, before any lesson:

Table 4 The 74 catalog questions with their arguments, with an empty learning store.

Measure	Value
questions	74
form chosen as the catalog expects	74
validated	63
not validated	10
gap (no row)	1
median time of the argument	624 ms
median SQL time	21 ms

The ten conclusions that are not validated are findings about the collections, and each names the work that would make it justifiable:

- *War and peace* (forms F1–F3, five questions): the period boundaries are conventional dates with the status *proposed*; peace is the complement of the war periods and inherits their status. Owner confirmation of the boundaries makes these conclusions validatable.
- *Speech acts read from a sentence* (E2–E4, E6, four questions): the precision of the sentence reading is too low for the acceptance grade; the learning campaign addressed this.
- *Mentions by surname* (B6 for Euler, one question): the 31 derivable instances for *Daniel Bernoulli* rest on the surname alone, which seven other entries of the Euler register share (J. Bernoulli, N. Bernoulli and others), and the Euler sidecar has no register index of the persons named in a letter that could serve as a second witness. For Kármán that index exists, and the same form is validated.

12 A retrieval baseline on the same questions

To see what the argument adds, the same 74 questions were answered by retrieval: the ten best regests and the ten best text paragraphs by BM25 on the question's words, passed with their ids to the same model with the instruction to answer from them and cite them (`scripts/fql_rag_baseline.py`). The baseline answers were compared with the arguments of the same questions by the counts they state.

Table 5 The retrieval baseline, compared with the arguments. A count is a digit run in the prose of the answer outside citation brackets, dates and years; an answer matches when any of its counts equals the figure.

Measure	Value
answers stating a count	31 of 74
questions whose argument establishes a count	62
of these, answers whose count equals the justified figure	3
of these, answers whose count equals the query’s figure	1
of these, answers stating counts that equal neither	24
lower-bound questions answered with <i>at least</i>	0 of 61
answers citing a passage	63 of 74

The comparison is lenient towards the baseline: an answer that mentions a matching number in passing counts as a match. Of the 62 questions whose argument establishes a count, the baseline answered 28 with a count; 24 of these state numbers that neither the query nor the argument establishes, and 34 give no count at all, often saying, as for Kármán’s correspondents, that the passages support only a few cases. None of the 61 lower-bound questions is answered with *at least*, while 63 of 74 answers cite passages. Citing a passage is not having a ground for a count: the passages are a sample of twenty texts, and no count over a collection of 15,976 letters follows from twenty texts. A first version of this measurement read digits inside the citation brackets (folder numbers) as counts and overstated the baseline’s counting answers; the figures above are from the corrected measurement.

13 Learning from unconvincing justifications

A reader who finds a justification unconvincing marks the node (a premise, the conclusion, an instance) and says why, choosing a category: a premise is missing or false, the inference does not hold, the conclusion is stronger or weaker than its grounds, persons are confused, a defeater was not considered, the scheme or the reading of the question is wrong, the evidence locus is wrong, the wording is poor. The annotation opens an **episode**, stored with the complete argument.

A learning Partitur [10] runs five stages. *Diagnose* (`gemini-3.8-flash`) assigns the error to one site: a rule, a critical question, a grade, a reliability, a form, an instruction of the harness, or the data, which is not learnable and is recorded as a data issue. *Generalise* writes a lesson as one general sentence and as a patch of operations on the declarative scheme file (add or change a critical question, add a rule, set a rule’s grade, set which measured sample governs a reading, change a form or a scheme text, change the acceptance thresholds) or on an instruction. *Propose* applies and checks the patch: it must parse, its probes must only read, a reliability may only name a measured sample, it must name no instance of the case (no letter id, register id, person or row value), and it must not equal a revoked version. *Gate* measures the baseline first, then the candidate, on the annotated case, on every stored case the patch can affect (an impact analysis over schemes, rules and critical questions), and on two mutations: invented witnesses and witnesses moved between mutually exclusive rows, neither of which may be accepted more often than before; a judge model decides whether the case is resolved and whether a changed case became worse. *Promote* activates the version when every check passes. Every learned version stays revisable; an owner revocation restores the previous version, and a revoked version is never activated again. Repeated gate failures of one category, or an owner revocation, open a **meta-episode** in which the lesson writer’s own instruction is rewritten and gated by re-running the failed episodes under it. Everything is recorded in one DuckDB file: episodes, stages, diagnoses, lessons, artifact versions, gate runs, replays, promotions, meta-episodes.

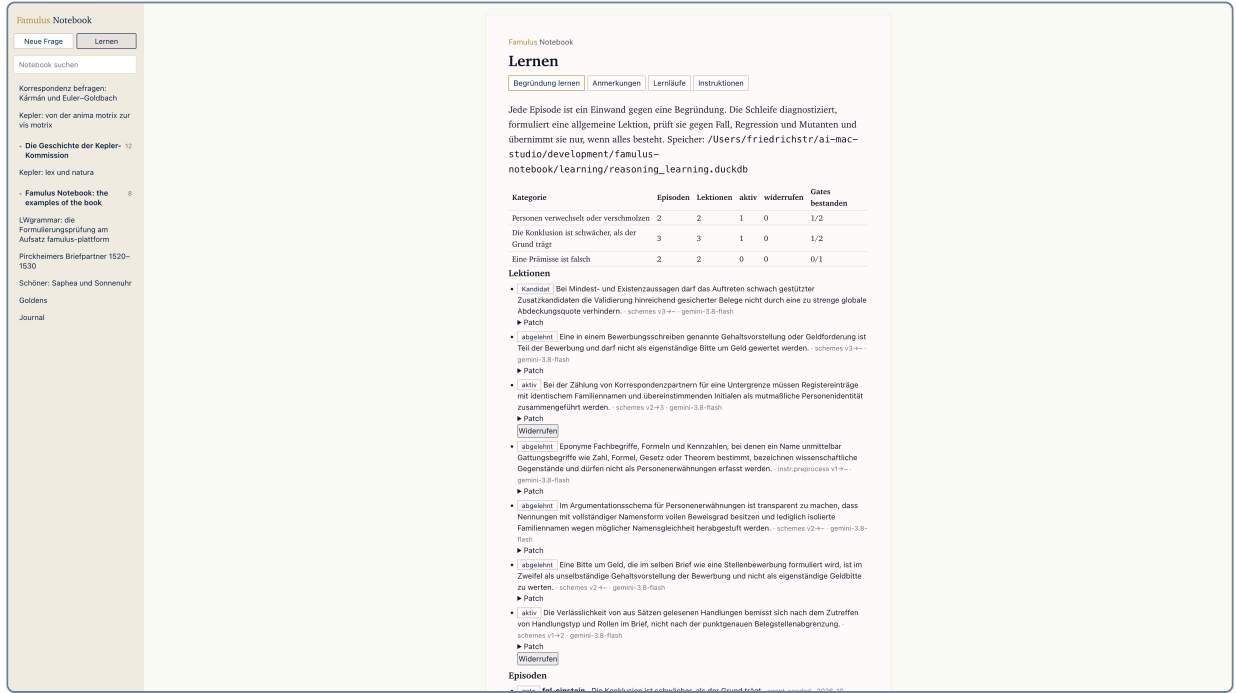


Figure 5 The review of the learning store: metrics by category, lessons with their patches and status, episodes with their diagnosis and gate reasons.

13.1 A first learning campaign

Seven episodes were run, from objections the agent derived from the measured audit findings; they are stored with the origin *agent-seeded*. Table 6 lists them; two re-runs follow a defect of the patch check that had refused a vocabulary word (*position*) as an instance of the case, fixed before the re-run.

Table 6 The learning campaign. The lessons are quoted in the German of the learner model.

Block	Objection	Lesson	Outcome
fql-geld	too weak	Die Verlässlichkeit von aus Sätzen gelesenen Handlungen bemisst sich nach dem Zutreffen von Handlungstyp und Rollen im Brief, nicht nach der punktgenauen Belegstellenabgrenzung.	learned
fql-stellen	premise wrong	Eine Bitte um Geld, die im selben Brief wie eine Stellenbewerbung formuliert wird, ist im Zweifel als unselbständige Gehaltsvorstellung der Bewerbung und nicht als eigenständige Geldbitte zu werten.	patch refused
fql-einstein	too weak	Im Argumentationsschema für Personenerwähnungen ist transparent zu machen, dass Nennungen mit vollständiger Namensform vollen Beweisgrad besitzen und lediglich isolierte Familiennamen wegen möglicher Namensgleichheit herabgestuft werden.	gate failed
fql-einstein-mit	persons confused	Eponyme Fachbegriffe, Formeln und Kennzahlen, bei denen ein Name unmittelbar Gattungsbegriffe wie Zahl, Formel, Gesetz oder Theorem bestimmt, bezeichnen wissenschaftliche Gegenstände und dürfen nicht als Personenerwähnungen erfasst werden.	gate failed
fql-partner	persons confused	Bei der Zählung von Korrespondenzpartnern für eine Untergrenze müssen Registereinträge mit identischem Familiennamen und übereinstimmenden Initialen als mutmaßliche Personenidentität zusammengeführt werden.	learned
fql-stellen	premise wrong	Eine in einem Bewerbungsschreiben genannte Gehaltsvorstellung oder Geldforderung ist Teil der Bewerbung und darf nicht als eigenständige Bitte um Geld gewertet werden.	gate failed
fql-einstein	too weak	Bei Mindest- und Existenzaussagen darf das Auftreten schwach gestützter Zusatzkandidaten die Validierung hinreichend gesicherter Belege nicht durch eine zu strenge globale Abdeckungsquote verhindern.	gate failed

2 of 7 episodes produced a lesson that passed every check. The first separates the reliability of a speech-act reading (type and roles, 27 of 30) from that of its evidence sentence (22 of 30) and lets the former weigh the act; its patch is a single line that names the measured sample. The second merges, for lower bounds, register entries with the same surname and compatible initials; the lower bound for Kármán's partners 1930–1940 drops from 489 to 441. Merging can only lower a lower bound, so the change is conservative. For Euler the same lesson takes the lower bound from 304 to 195: the register of IVA writes members of one family under one surname with compatible initials, so the Bernoullis and other families collapse into fewer partners. The figure stays a true lower bound, but it is less informative; the lesson is therefore marked for the owner's review rather than treated as settled.

The gate rejected 5 lessons, and for reasons that matter: a lesson that only rewrote a scheme's text left the case unchanged; a lesson that appended a rule to an instruction of the harness could not change an argument already built; and a lesson that lowered the global coverage threshold to make a weak case pass was judged to make another case worse, because it validated an argument whose witnesses mostly do not carry. Two

failures in one category opened meta-episodes; both rewrote the lesson writer’s instruction, and neither candidate made a failed episode pass on replay, so neither was adopted.

The two episodes on the Einstein mentions failed for a second reason, found only afterwards: their gates ran under the prover defect described in Section 5, which let a critical question act on only the first 200 of its 1,302 rows. The objection was right, the remedy was already in the scheme, and after the correction the case is validated without a lesson. The episodes keep their verdict and carry a note in the learning store; a gate is only as good as the determinism of the replay it compares.

Table 7 The 74 catalog questions with an empty learning store and with the learned lessons: same code, same sidecars, same questions.

	without lessons	with lessons
validated	63	66
not validated	10	7
gap	1	1

The bench runs the language model again, and the model does not read every question the same way twice: it may bind a name to another set of register entries or choose other parameters. The effect of the lessons alone is therefore measured by replay (`scripts/fql_lesson_effect.py`): every argument of the bench is rebuilt on the same rows and the same bound parameters, once with the shipped schemes and once with the learned ones. Of 74 arguments, 10 change:

Table 8 The effect of the lessons by replay: same rows, same bindings, shipped against learned schemes. The figure is the justified count of the conclusion.

Form	Question	without lessons	with lessons
C1 (Euler)	<i>Mit wem korrespondierte Euler?</i>	validated, 304	validated, 195
C1 (Kármán)	<i>Mit wem korrespondierte Kármán 1930–1940?</i>	validated, 489	validated, 441
C3 (Kármán)	<i>Wer schrieb an Kármán 1945?</i>	validated, 34	validated, 32
E1 (Kármán)	<i>Wer bat Kármán um eine Stelle?</i>	validated, 36	validated, 51
E2 (Kármán)	<i>Wer bat Kármán um finanzielle Unterstützung?</i>	not validated, 26	validated, 73
E3 (Kármán)	<i>Wie viel Geld sagte Kármán anderen zu?</i>	not validated, 2	validated, 14
E4 (Kármán)	<i>Wen empfahl Kármán?</i>	not validated, 91	validated, 210
E5 (Kármán)	<i>Wer bat Kármán um Sonderdrucke?</i>	validated, 228	validated, 428
E6 (Kármán)	<i>Wie entwickelten sich die genannten Geldsummen über die Jahre?</i>	not validated, 10	validated, 85
H2 (Kármán)	<i>Wer bat Kármán um eine Stelle, und antwortete Kármán?</i>	validated, 36	validated, 55

14 Related work

The schemes and their critical questions take their form from informal logic [4], [5]; the treatment of defeaters as reasons that can withdraw a conclusion follows the theory of defeasible reasoning [11], and the acceptance of conclusions under attack has a formal model in abstract argumentation [12]. What is particular here is the ground: the facts are rows of a scholarly collection with addresses down to the page and the sentence, the defeaters are queries over the same rows, and the strength of a model reading is its measured

precision. The search for a justification as an expansion of a goal over declared rules, and the reading of the closed expansion as the proof, are the familiar treatment of Horn-clause reasoning (goal-directed resolution over a rule base); what is added is the declared map of grounds that closes every branch in a document, a measured reading or a stipulation, and the reduction of the proof to a core in which no node is idle. The relation to retrieval-augmented generation [1] is the inversion described in Section 2. The companion papers describe the platform [13], the notebook [14], the score language [10], the golden queries that test the machinery on questions with known answers [15], the lexical institution that governs the terms [16], and the epistemological account on which the verdict rule rests [2].

15 Limits

- Validation is justification relative to the registered ground, not a guarantee of truth. If a register entry is false, every argument that rests on it is unsound: its conclusion loses this justification, and the system can name exactly the conclusions concerned, because every premise carries its address. The conclusions are not thereby false — from the falsity of a premise nothing follows about the truth of the conclusion. Correcting the entry and re-running the argument restores or revises the verdict.
- The precisions of the model readings rest on samples of 20 to 44 items, read by the agent that built the sidecar, not by the owner. The seeded annotations of the learning campaign are the agent's, not the owner's.
- Closure is declared for no collection; every count is a lower bound and no absence follows.
- One learned lesson merges register entries by surname and initials for lower bounds. It is conservative for a lower bound, since merging can only lower the count, but it compares name forms by their parts; it is marked for the owner's review.
- The grounding is per witness; arguments with more than 20,000 instances are cut, and the uncounted instances are reported as a gap.
- The minimal core is reduced under a time budget. An instance whose reduction did not finish is marked unchecked and is shown with the leaves of its first proof; the tree then may contain nodes that a reduction would remove. The tree of a form is aggregated over instances, so it shows which rules and grounds carry and in how many instances; it does not show each instance's own proof, which the expanded view lists.
- That the expansion ends in grounds is a property of the declared map of grounds. A predicate missing from the map has no stock and is reported as a gap; a ground that is declared but not true of the data (a register entry that does not say what the rule needs) is what the critical questions and the annotation are for.
- The link from an evidence card into the edition viewer is a gap for the Euler volume IVA 4 until the volume is registered there; the notebook shows the facsimile itself.

16 Conclusion

An answer of Famulus is the conclusion of an argument whose premises are addressed facts of the collection, whose rules are declared, and whose defeaters were searched in the same data before any sentence was written. The argument is the result of a goal expansion that ends only in grounds, reduced to a core in which every node carries, and it is shown as a tree the reader can walk from the conclusion to the documents. The argument certifies a justification, not a truth: the conclusion follows from premises of a measured strength, and every premise can be checked at its address. The quantifier of the answer is what the grounds entail; for registers without declared closure, every count is a lower bound, and the answer says so. What cannot be justified is reported as not validated, with its weakest premise, and becomes a task: confirm a boundary, measure a reading, add a witness. An annotation of an unconvincing justification becomes a lesson for all arguments of its kind, gated against the case, the collection and invented evidence, and revocable by the owner.

Bibliography

- [1] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

- [2] G. Graßhoff, “Computational Epistemology: Foundations. Evidence and Its Justificatory Reach in the Sciences and the Humanities,” 2026.
- [3] M. Raasveldt and H. Mühleisen, “DuckDB: an Embeddable Analytical Database,” in *Proceedings of the 2019 International Conference on Management of Data (SIGMOD '19)*, ACM, 2019, pp. 1981–1984. doi: 10.1145/3299869.3320212.
- [4] D. Walton, *Argumentation Schemes for Presumptive Reasoning*. Mahwah, NJ: Lawrence Erlbaum Associates, 1996.
- [5] S. E. Toulmin, *The Uses of Argument*. Cambridge: Cambridge University Press, 1958.
- [6] M. W. Moskewicz, C. F. Madigan, Y. Zhao, L. Zhang, and S. Malik, “Chaff: Engineering an Efficient SAT Solver,” in *Proceedings of the 38th Design Automation Conference*, Jun. 2001, pp. 530–535. doi: 10.1109/dac.2001.935565.
- [7] W. F. Dowling and J. H. Gallier, “Linear-Time Algorithms for Testing the Satisfiability of Propositional Horn Formulae,” *The Journal of Logic Programming*, vol. 1, no. 3, pp. 267–284, 1984, doi: 10.1016/0743-1066(84)90014-1.
- [8] M. Davis, G. Logemann, and D. Loveland, “A Machine Program for Theorem-Proving,” *Communications of the ACM*, vol. 5, no. 7, pp. 394–397, 1962, doi: 10.1145/368273.368557.
- [9] J. A. Robinson, “A Machine-Oriented Logic Based on the Resolution Principle,” *Journal of the ACM*, vol. 12, no. 1, pp. 23–41, 1965, doi: 10.1145/321250.321253.
- [10] G. Graßhoff, “The AI Partitur — A Score Language for the Humanities,” 2026.
- [11] J. L. Pollock, “Defeasible Reasoning,” *Cognitive Science*, vol. 11, no. 4, pp. 481–518, 1987, doi: 10.1207/s15516709cog1104_4.
- [12] P. M. Dung, “On the Acceptability of Arguments and its Fundamental Role in Nonmonotonic Reasoning, Logic Programming and n-Person Games,” *Artificial Intelligence*, vol. 77, no. 2, pp. 321–357, 1995, doi: 10.1016/0004-3702(94)00041-X.
- [13] G. Graßhoff, “Famulus. Eine verteilte lokale Plattform für professionelle Forschung,” 2026.
- [14] G. Graßhoff, “Famulus Notebook,” 2026.
- [15] G. Graßhoff, “Golden Queries in AI Erasmus: A Testbed for Scholarly Inquiry,” 2026.
- [16] G. Graßhoff, “Wittgensteinian Grammar and OED Dictionaries for AI: A Foundation of Semantic Understanding of Scientific Texts,” *arXiv preprint*, 2026.