CHAPTER

Task 2 -- Causal coding -- minimalist style

CONTENTS

A formalisation of causal mapping

> Standing on the shoulders of giants

> In this chapter we present some of key general principles about how to do causal mapping which we at Causal Map Ltd (and, most of the time, at BathSDR) have adopted.

This is a very restricted yet powerful **minimalist** approach which we have also called "barefoot" or "naïve" coding.

In the next chapter Tasks 2 & 3 -- Introduction we look at specific conventions to make causal coding simple and powerful.

A formalisation of causal mapping

Thanks for inviting me to the discussion, James. **I'll start by describing the "minimalist" approach** to coding causal statements used for QuIP and developed originally by James, Fiona and colleagues at BathSDR and developed and further formalised at Causal Map Ltd in collaboration with BathSDR. This formalisation lives inside the Causal Map app. Then I will try to **answer the question of whether it can help us deal with more complicated constructions like enabling and blocking** and whether this could help us with mid-range theory. As an appendix I'll add a more detailed overview of minimalist causal coding.

The minimalist approach is notable because it is based in our **joint experience of coding thousands and thousands of stakeholder interviews and other data such as project reports**, mostly from international development and related sectors, as well as coding hundreds of thousands of pages with AI-assisted coding. These have nearly always involved **multiple sources talking about at least partially overlapping subject matter**. So this coding produces individual causal maps for each source, which can then be combined in various ways -- rather than constructing single-source maps of expert thinking (Axelrod, 1976) or the collective construction of a consensus map (Barbrook-Johnson & Penn, 2022).

Our experience has been that the vast majority of causal claims in these kinds of texts are easily and satisfactorily coded in the simplest possible form "X causally influenced Y". Explicit invocation of concepts like enabling/blocking, or necessary and/or sufficient conditions, or linear or even non-linear functions, or packages of causes, or even the strength of a link, are relatively rare. The causes and effects are not conceived of as variables, the causal link is undifferentiated, without even polarity, and if any counterfactual is implied it remains very unclear.

This approach is what we call **"Minimalist" or "Barefoot" Coding**.

So what? Can we use minimalist coding to code and make deductions about, say, enablers and blockers?

Using more sophisticated, non-minimalist coding such as DAGs or fuzzy cognitive maps or whatever allows one to code linear or even non-linear causal influences of single or even multiple causes on their effects. One can do the "coding" by simply writing down (using appropriate special syntax) the connections, because one is an expert, and/or one can verify such statements statistically on the basis of observational data. Thus armed, one can make predictions or have sophisticated arguments about counterfactuals. But using minimalist coding we cannot do that, because our claims are formally weaker and therefore our inference rules are weaker. What we *can* do is still really interesting. We can ask and answer useful questions like:

- what are the main influences on (or effects of) a particular factor, according to the sources?
- what are the upstream, indirect influences on (or effects of) a particular factor, bearing in mind The transitivity trap?
- how well is a given programme theory validated by the respondents' narratives? (We can do this basically by using embeddings to get measures of semantic similarity between labels and

aggregate these as a goodness of fit of theory to data.)

That is all exciting and useful. It's a surprisingly simple way to make a lot of sense out of a lot of texts which is, with caveats, almost completely automatable, **but** James suggests that maybe we could ascend from formally weaker but numerically overwhelming minimalist-coded data to make other rich conclusions, in particular about enablers and blockers like the headphones and the rain. However, I don't think this is really possible. In minimalist coding, at the level of individual claims, you can code "The headphones enabled James to answer the question in the Zoom call" as

> The headphones --> James was able to answer the question in the Zoom call

... but we cannot easily get inside the *contents* of the effect. We might like to be able to code this as the effect of the headphones not on a simple causal factor but on *another causal connection*, namely between the question on the Zoom call and James' answer, but we do not have any way at the moment to do this. It might be possible to extend minimalist coding to cope with this, perhaps ending up with three factors (headphones, question, answer) and some new syntactic rules to code their relationship, and some corresponding new semantic rules to be able to deduce more things about these three factors, but I think this would be **missing the point**. I'm not sure what we could do with these kinds of subtle relationships at any scale. Let's guess that within a given corpus, five percent of causal claims are of this form: what are the chances of such claims then overlapping enough in content that we could then apply our new more specialised deduction rules in more than a handful of cases?

It might be the case that certain specific more sophisticated causal constructions become **part of ordinary language**. For example: "Her post mocking Farage went viral, so Farage was forced to respond". Here, the concept of *going viral* is perhaps a kind of shorthand for a quite sophisticated causal claim, yet it might be common enough for us to be able to usefully code it (and reason with it) using only unadulterated minimalist coding, without causally unpacking "her post went viral". So that's useful, and maybe it is even useful in building some kinds of mid-range theory, but without actually understanding or unpacking what "going viral" means.

So that's it, in a nutshell. Sorry to disappoint, James.

Appendix: Minimalist coding

The 90% rule

We have found that it is pretty easy to agree how to apply minimalist coding to say 90% of explicit causal claims in texts, without missing out essential causal information, whereas it is very difficult to find appropriate frameworks to cope with the remaining 10%.

Fewest assumptions

Minimalist coding is perhaps the most primitive possible form of causal coding which makes no assumptions about the ontological form of the causal factors involved (the "causes" and "effects") or about *how* causes influence effects. In particular we do not have to decide if cause and/or effect is perhaps Boolean or ordinal, or if perhaps multiple causes belong in some kind of package or if there is some kind of specific functional relationship between causes and effects.

An act of causal coding is simply adding a link to a database or list of links: a link consists of **one new or reused cause label and one new or reused effect label**, together with the highlighted quote and the ID of the source.

A statement (S) from source Steve:

> I drank a lot and so I got super happy
>
> can be trivially coded minimalist-style as
>
> I drank a lot --> I got super happy (Source ID: Steve; Quote: I drank a lot and so I got super happy)

That's it.

Causal maps

Crucially, we can then display the coded claims for individuals as a graphical causal map, and we can also display the entire map for all individuals and/or maps filtered in different ways to answer different questions. There is a handful of other applications (Ackermann et al., 1996) (Laukkanen, 2012) for causal mapping which also do this; but as far as we know, only Causal Map also allows direct QDA-style causal coding of texts.

Data structure

Although we have the option of creating additional tags associated with each link (where many approaches would for example code the polarity of a link) this is not central to our approach.

We don't use a separate native table for factor labels: they are simply derived on the fly from whatever labels happen to be used in the current table of links. This makes data processing simpler and also suggests an ontological stance: causal factors only exist in virtue of being part of causal claims.

We do however have an additional table for source metadata including the IDs of sources, which can be joined to the links table in order, for example, to be able to say "show me all the claims made by women".

Causal powers

We adopt an explicitly realist understanding of causation, because we think that's what people mean. The outcome occurred in virtue of the causal powers: drinking a lot causally influenced the super happiness in virtue of its causal powers to do so; that's what makes it a causal claim rather than just a remark on a co-occurrence or a sequence of events.

Causal influence, not determination

We believe that it's rare for people to make claims about causal determination: someone can say that the heavy drinking made them super happy and then also agree that the music had a lot to do with it too, without this feeling like a contradiction.

Not even polarity

We differ even from most other approaches which are explicitly called "causal mapping" in that we do not even "out of the box" record polarity of links (to do so would involve making assumptions about the nature of the "variables" at each end of the link as well the function from one to the other).

The Focus on Cognition

In the minimalist approach, **we are quite clear that what we are trying to code is the speaker's surface cognitions and causal thinking**, while the actual reality of the things themselves is simply bracketed off at this stage, either to be revisited later (because we are indeed interested in the facts beyond the claims) or not (because we are anyway interested in the cognitions).

Staying on the surface

At Causal Map, we rarely make any effort to get beneath the surface, to try to infer hidden or implicit meanings. This is particularly well-suited to coding at scale and/or with AI. Our colleagues at BathSDR do this a bit differently, spending more effort to read across an entire source to work out what the source *really* meant to say.

Closer to the cognitive truth

It's really easy to code statements like (S) using minimalist coding. The trouble with trying to use more sophisticated frameworks is that they are nearly always *ontologically under-determined*. For example, even a simple approach like Causal Loop Diagramming is strongly functional and requires at least a monotonic relationship between the variables: something like, *the more* I drink, *the happier* I get (in addition to which we have to code the actual, factual claims: I did drink a lot, and: I did get super happy). But is that what the speaker meant? How do we know if the speaker has say a continuous or Boolean model of "drinking"? If Boolean, what is the opposite of drinking a lot? Drinking only a little? If continuous, how do we know what kind of function they use in their own internal model?

We'd say: nonsense. To code most causal claims as meaning some functional relationship between variables is mostly over-specified and psychologically wrong. Trying to apply such non-minimalist models means that even the trivially easy 90% of causal coding becomes suddenly hard. Of course, you can just declare that we are going to use a particular kind of non-minimalist coding for everything, but which? If we code "I got really tired because I have Long Covid", we could perhaps code both cause and effect as Boolean variables, but what about "I got really tired because it was really hot", and "I got really tired because it was really cold" how are we going to code "it was really hot" and "it was really cold"? Is there a moderate temperature which does not have this effect? How moderate? Does this variable pass through zero and come out the other side into minus temperatures? ((Ragin, 2008)) If what want to do is model a system, we can pick any solution we want. But if we want to model *cognition*, any of these answers is usually over-specified.

Unclear counterfactuals

More formal, non-minimalist coding has clear counterfactuals. These may often be Boolean or continuous (the volume depends on the position of the volume dial; it's a 10, so the volume is maximum, if it had been at 5, the volume would have been about half as loud, and so on). Minimalist coding arguably implies some kind of naked counterfactual, but it is not always clear exactly *what*.

General versus specific

Minimalist coding focuses primarily on **factual causal claims** which also warrant the inference that both X and Y actually happened / were the case.

Most causal claims in the kinds of texts we have dealt with (interviews and published or internal reports in international development and some other sectors) are factual, about the present or past. Sometimes we see general claims, and we often just code these willy-nilly. In any case, the distinction between general claims and claims about specific events that actually happen is often fractal and difficult to maintain completely when modelling ordinary language.

Minimalist coding as "qualitative causal" coding

Minimalist coding may be reasonably also called **Qualitative Causal Coding**. It shares characteristics with some forms of coding within Qualitative Data Analysis (QDA), in particular demonstrating an asymmetry between presence and absence.

We don't code absences

We do not code absences unless they are specified within the text. While codes may be counted, the concept of a *proportion* of codes is challenging because the denominator is often unclear. So if families are talking about reasons for family disputes, and family F mentions social media use, and family G mentions homework, we do not usually assume that family F does *not* think that homework can also be a cause of family disputes.

The labels do all the work

At Causal Map Ltd, our canonical methodology initially involves in vivo coding, using the actual words in the text as factor labels. This initial process generates hundreds of overlapping factor labels. This part is really easy (and is easy to automate with AI). Obviously, hundreds (or hundreds of thousands) of overlapping factor labels are not very useful, so we need to somehow consolidate them. Arguably, minimalist coding makes the initial coding easy but it just defers some of the challenges to the recoding phase. We can:

- Use human- or AI-powered clustering techniques to consolidate the codes according to some theory
- Use AI-powered clustering techniques to consolidate the codes according to automated, numerical encoding of their meanings
- "Hard-recode" the entire dataset using a newly agreed codebook (see above)
- "Soft-recode" the dataset on the fly using embeddings to recode raw labels into those codebook labels to which they are most similar

None of this really answers all the questions raised above about problematic cases such as what to do with "I got really tired because it was really hot", and "I got really tired because it was really cold" or any other case where we have different factor codes which have shared information. At first blush, this isn't a problem, we can simply code "it was really hot" and "it was really cold" separately, but how to parse the contents to reflect the fact that these two are related? Or, how to parse the contents of "Improved health behaviour (hand washing)" and "Improved health behaviour (using a mosquito net)" to reflect the fact that they are somehow neighbours? We do have some tricks for this, but that would take us beyond the present discussion.

See also:

(Powell et al., 2024)

(Powell & Cabral, 2025)

(Britt et al., 2025)

(Powell et al., 2025)

(Remnant et al., 2025)

References

Ackermann, Jones, Sweeney, & Eden (1996). *Decision Explorer: User Guide*. https://banxia.com/pdf/de/DEGuide.pdf.

Axelrod (1976). *Structure of Decision: The Cognitive Maps of Political Elites*. Princeton university press.

Barbrook-Johnson, & Penn (2022). *Participatory Systems Mapping*. In *Systems Mapping: How to Build and Use Causal Models of Systems*. https://doi.org/10.1007/978-3-031-01919-7_5.

Britt, Powell, & Cabral (2025). *Strengthening Outcome Harvesting with AI-assisted Causal Mapping*. https://5a867cea-2d96-4383-acf1-7bc3d406cdeb.usrfiles.com/ugd/5a867c_ad000813c80747baa85c7bd5ffaf0442.pdf.

Laukkanen (2012). *Comparative Causal Mapping and CMAP3 Software in Qualitative Studies*. https://doi.org/10.17169/fqs-13.2.1846.

Powell, Copestake, & Remnant (2024). *Causal Mapping for Evaluators*. https://doi.org/10.1177/13563890231196601.

Powell, & Cabral (2025). *AI-assisted Causal Mapping: A Validation Study*. Routledge. https://www.tandfonline.com/doi/abs/10.1080/13645579.2025.2591157.

Powell, Cabral, & Mishan (2025). *A Workflow for Collecting and Understanding Stories at Scale, Supported by Artificial Intelligence*. SAGE PublicationsSage UK: London, England. https://doi.org/10.1177/13563890251328640.

Ragin (2008). *Measurement Versus Calibration: A Set-Theoretic Approach*. https://doi.org/10.1093/oxfordhb/9780199286546.003.0008.

Remnant, Copestake, Powell, & Channon (2025). *Qualitative Causal Mapping in Evaluations*. In *Handbook of Health Services Evaluation: Theories, Methods and Innovative Practices*. https://doi.org/10.1007/978-3-031-87869-5_12.

A formalisation of causal mapping

Why we stick to bare causation in causal mapping.

Our rule of thumb: record only that “C causes D.” No coding of necessity, non-linearity, moderators, or strength. Just who said what causes what.

The short case

- **It avoids false precision.** Labelling links as “necessary,” “moderator,” “non-linear,” or assigning strengths suggests evidence we rarely have. We prefer to show what was claimed and how often, then let readers judge. Maps are primarily **epistemic**—repositories of evidence about people’s beliefs—not truth machines.
- **It scales and compares.** Bare links plus rich factor labels let us aggregate, filter, and compare across sources, groups, and contexts without fighting about semantics of special symbols. Our tools then summarise with counts (citations, sources) and simple derived measures (like “outcomeness”), instead of speculative link attributes.

What we record

- **Factors (boxes):** short propositions that do the heavy lifting (e.g., “Not enough money,” “Won’t take a holiday this year”).
- **Links (arrows):** undifferentiated causal influence claims between factors. A link means “P said C influences D.” That’s it.

What we deliberately don’t code on links

- Necessity/sufficiency
- Non-linear forms or feedback classifications
- Moderator/mediator/inhibitor role
- Polarity or strength

Why? Because (a) respondents seldom state these explicitly; (b) analysts rarely agree on them from text alone; and (c) they reduce inter-coder reliability and slow projects down without very much which we can dependably aggregate.

Our analyses are still useful

Coding bare links doesn’t make maps “impoverished”: Causal mapping produces models you can query to answer questions

Bottom line

Most of the time, we code only: “C causes D (as claimed by P).” That minimal, transparent unit is reliable, scalable, and faithful to the data people actually provide. Everything richer belongs in **analysis and interpretation**, not in speculative link types baked into the coding.

Our approach clearly distinguishes evidence from facts and does not automatically warrant causal inferences

Our approach is minimalist -- factors are not variables

Many or most causal mapping approaches, including Causal Loop Diagrams, also code the perceived strength of a causal link. This means that the factors become variables which can take values between, say, low and high or positive and negative, and we can make a much broader range of inferences using some form of numerical modelling. This can be seen as the extreme reproducible end of our spectrum and borders on quantitative approaches.

However we do not go so far: our causal factors are closer to being propositions rather than variables and we do not jump to code, say, poverty as negative wealth, or unemployment as obviously just the opposite of employment.

The Conventional Assumption

A foundational assumption, particularly for those approaching causal mapping from a systems dynamics perspective, is that every concept on a map should be treated as a **variable**. This implies that each element is something quantifiable, capable of taking on different values across a defined spectrum, such as from low to high, negative to positive, or from zero upwards. Such a map is backed up by a dataset, a large-ish set of measurements of the state of each variable.

The Discrepancy with Human Narrative

However, this assumption contrasts sharply with how people actually communicate and describe their experiences. When individuals explain what causes what in their world, they rarely speak in terms of discrete variables. Forcing real-world narratives into a rigid, variable-based structure requires significant and often unnatural contortions.

Constructing variables out of experience is just that: a construction. Quantitative social scientists are really good at it. But people's thinking and language are not inherently structured in this way.

For instance, in an evaluation of a program's effects, the sudden onset of the COVID-19 pandemic presents a significant modelling problem. While the pandemic certainly had a causal impact on countless factors, it doesn't fit neatly into the definition of a variable.

How would one define it? As a binary "COVID vs. no COVID" variable? The concept of a counterfactual -- a world where the pandemic never happened -- is abstract and difficult to operationalize. This example highlights that the way people experience and discuss the world is often event-based, not variable-based, exposing a limitation in traditional modelling assumptions.

1a A minimalist approach to coding helps capture what people actually say

1a A minimalist approach to coding helps capture what people actually say

We just argued that 1a A minimalist approach to coding helps capture what people actually say. But even if you did succeed in imposing some special logical features on your data -- for example, coding necessity and sufficiency -- you'd probably find that most of your data didn't fit well with these special features. When it comes to aggregating medium or large amounts of coding, you wouldn't find it very useful.

With our minimalist approach, we mostly have just one task: what to do about all those different factor labels.

1c A minimalist approach to coding does not code absences

One thing which makes causal mapping a fundamentally qualitative approach is that we do not code absences.

We do not think that the world, nor the piece of the world we are studying, is essentially a grid of variables and cases (nor a cube of variables and cases and timepoints), in which each case always has a value for every variable (at every timepoint).

If some respondents say that their headaches make them nauseous, and others do not mention headaches, even if they mention nausea, we do not interpret that as meaning that they *did or did not* have headaches. We do not think that having headaches, or not, is a variable which *must* be relevant to everyone's explanations, all the time.

Our approach is minimalist -- we do not code the strength of a link

In a causal mapping dataset there is no need for a special table of factors

Factor labels -- a creative challenge

Where do the labels for the causal factors come from? As with ordinary QDA and thematic analysis (Braun and Clarke, 2006), approaches vary in the extent to which they are purely exploratory or seek to confirm prior theory (Copestake, 2014). Exploratory coding entails trying to identify different causal claims embedded in what people say, creating factor labels inductively and iteratively from the narrative data. Different respondents will not, of course, always use precisely the same phrases, and it is a creative challenge to create and curate this list of causal factors. For example, if Alice says 'Feeling good about the future is one thing that increases your wellbeing', is this element 'Feeling good about the future' the same as 'Being confident about tomorrow' which Bob mentioned earlier? Should we encode them both as the same thing, and if so, what shall we call it? We might choose 'Positive view of future', but how well does this cover both cases? Laukkanen (1994) discusses strategies for finding common vocabularies. As in ordinary QDA, analysts will usually find themselves generating an ever-growing list of factors and will need to continually consider how to consolidate it – sometimes using strategies such as hierarchical coding or 'nesting' factors (as discussed in the following section).

The alternative to exploratory coding is confirmatory coding, which employs an agreed code book, derived from a ToC and/or from prior studies. QuIP studies mostly use exploratory coding but sometimes supplement labels with additional codes derived from a project's ToC, for example, 'attribution coding' helps to signify which factors explicitly refer to a specific intervention being evaluated (Copestake et al., 2019b: 257). However, careful sequencing matters here because pre-set codes may frame or bias how the coder sees the data (Copestake et al., 2019a). Again, the positionality of the coder matters just as much when doing causal coding as it does for any other form of qualitative data coding.

Factor label tags -- coding factor metadata within its label

For example you might want to code the respondent's happiness at work as different from yet similar to their happiness at home. With a factor table, you could have a field called `label` = "Happiness" and another, say `context`, which is = either "Home" or "Work". This is what we do with the links table in Causal Map, where we do have some hard-coded (but optional) fields and some user-definable fields.

Hierarchical coding is one way to bring some order to a whole crowd of factors. However, sometimes you don’t want to think in terms of a strict hierarchy, or maybe you have an additional set of themes which cut across that hierarchy.

https://vimeo.com/671894620

Tags are useful in either of these cases.

Tags are just sequences of characters within a factor label to which you have given a special meaning, and which are unique and easy to search for. These can include letters, emojis or phrases. You can do coding without any such tags if you want, but it can help when searching and filtering.

Factor tags are just like #️⃣ Link hashtags. Confusingly, a link hashtag doesn’t have to actually start with a `#`, and a factor tag can indeed start with a `#`, but we find it easier to keep the names separate like this.

So a tag is nothing more than any sequence of characters which is repeated in several factor labels. Any sequence of characters will do. For example you could consider the letter “a” to be a tag and display the map showing all the factors which contain the letter “a”. But this wouldn’t be interesting. The trick when using tags is to decide on short, meaningful codes which will not be repeated anywhere else. For example you wouldn’t want to use a pair of tags like “women” and “men” to distinguish factors which are only relevant for one or the other gender because the “wo**men**” factors would also turn up when you search for “**men**”. That is why we have to be careful when creating tags, for example by preceding a sequence of characters with a tag `"#"`.

A quote like “family situation is better now because of improved food availability” can be coded like this:

> More food –> Improved wellbeing

Now, maybe you are asked also to keep track of any aspects of the project which have to do with nutrition. Nutrition is not really part of your system of factors, but you would like to be able to construct some maps just to look at this aspect. So you can write this:

> More food #nutrition –> Improved wellbeing

Similarly, if Improved Wellbeing is one of the desired outcomes of the project, we might want to reflect that by adding a tag “(Outcome)” like this.

> More food –> Improved wellbeing (Outcome)

Then we can easily search for this and other desired outcomes.

A tag like “men” is not suitable because it is likely to appear elsewhere (e.g. as part of “women” or “management”). To get round this, add additional characters like a hash: “#men”; this makes the tag unique.

If you use curved or square brackets around your tags, you can use one of the app filters to hide the tags for specific maps if desired.

In a causal mapping dataset there is no need for a special table of factors

It might be tempting to try to formulate all factor labels in a strictly similar way, using for example language like increased probability of … or positive change in … in every case. But it is difficult to identify and agree on a satisfactory template for doing this which will capture enough of the way people really make causal explanations (in the way that quantitative social scientists hope to measure everything just with continuous variables). This is always a balancing act, but we encourage you when in doubt to stick fairly close to the actual language your sources use (so-called "in-vivo" coding), and don't be *too* worried if your factor labels are different from one another grammatically (e.g. some express a difference like improvement in X and some do not).

The formulation of **factor labels** should fit the intended interpretation of the **causal links**. For example, most commonly B ➜ E is supposed to mean that B exerts in some sense an "increasing" or "decreasing" influence on E, then both B and E need to be formulated in a corresponding way. In order to ease interpretation, with a few exceptions, factors should be labelled and understood in such a way that it makes sense to say "more of this" or "this happened as opposed to not happening": we call these semi-quantitative factors.

Consequently you should avoid a factor label like Training courses, which might be understood as a mixed bag of various causal factors to do with training courses. We would usually prefer a label such as Training courses delivered or Quality of training courses which are easier to understand as things which can increase or decrease, or happen or not happen. You may even prefer to use labels like Quality of training courses improved or Improved quality of training courses, in which the *difference made* is already included in the title.

Examples of semi-quantitative factors

These are examples of factor labels where you can judge whether it happened more or less, whether it is higher or lower, or whether it happened versus not happened:

- Sold cow
- Earthquake happened
- (Had) good harvest
- (Level of) bank account
- (Level of) ethnic tolerance
- Quality of seeds

In some contexts, we can also talk about the *likelihood* of events, so "if people get a good harvest they are less likely to sell their cow."

Non-quantitative factors

It is also perfectly acceptable and sometimes necessary to use purely qualitative labels, e.g. coping style, see below. However, this may limit some of the analysis and reporting tools available:

- Teaching style

- Coping strategy
- The content of the report

We can even make a link between two such factors, claiming for example that the style of 60's music influenced the style of 70's music, without any concept of quantity. That's ok.

Factor labels -- a creative challenge

In a causal mapping dataset there is no need for a special table of factors

Factor labels -- do not over-generalise

When you are creating factor labels for re-use across different causal claims, you should usually take care to keep them specific: make them no more general than they need to be.

So if you are coding cases where a household has increased income, use a label like Increased household income, not Increased income or even Economic improvement.

This is especially important when using hierarchical factors, when it’s easy to fall into the temptation of creating very general top-level labels like Economic improvement even if all your material is actually only about increased income in households and farms.

Our approach is minimalist -- we do not code the strength of a link

In our implementation of causal mapping in the Causal Map app, Our approach is minimalist -- we do not code the strength of a link.

Providing metadata as a column makes sense when the values of this column make sense across the whole dataset, across all multiple links, like let's say before covid and after covid.

Such a column can function a bit like a *context* variable, for different time periods or applying to different stakeholders. Context in this sense might be seen as functioning a *bit* like a causal factor but not exactly.

But we can also provide metadata as free-form tags. We provide a hard-coded "tags" column for which users can provide comma-separated lists of tags which are made up and adapted on the fly. They don't necessarily make sense across the whole dataset.

In Causal Map 4, as well as a hard-coded Tags column, we do provide a hard-coded sentiment column which can take the values -1, 0 and 1, and which can be averaged to any number between -1 and 1.

Link metadata -- Sentiment ▸

We also provide arbitrary additional free-form, free-text columns for any purpose. We often like to add a column like this:

Link metadata -- Time reference ▸

Link metadata -- quality of evidence ▸

... or simply to code a tag like "#doubtful".

Link metadata -- Sentiment

What is it for?

a hard-coded sentiment column which can take the values -1, 0 and 1, and which can be averaged to any number between -1 and 1.

Link metadata -- Time reference

It is often useful to code a time reference. We often conflate time with hypothetical status, e.g.

- hypothetical past/present
- factual-past/present
- future-planned
- future-hypothetical

For example, if we are to code a whole corpus of reports which also include planning documentation, there might be a lot of causal claims about what is supposed to happen in the future, perhaps interspersed with claims about what actually happened in the past. It will often be important to distinguish these two.

Link metadata -- quality of evidence

Research on LLMs' ability to detect causal claims

The Semantic Engine of Cause: Tracing the Emergence of Informal Causal Understanding in Large Language Models (2015–2025)

1. Introduction: The Emergence of "Native" Causal Fluency

The capacity of Large Language Models (LLMs) to identify, generate, and reason about causal relationships in ordinary language represents one of the most significant, yet enigmatic, developments in artificial intelligence over the last decade. As noted by observers of models since the release of ChatGPT (based on GPT-3.5) and its successors, these systems exhibit a "native" ability to process prompts involving influence, consequence, and mechanism without requiring the extensive few-shot examples or rigid schema engineering that characterized previous generations of Natural Language Processing (NLP). This report investigates the trajectory of this capability from 2015 to 2025, deconstructing whether this proficiency is a serendipitous artifact of scale or the result of specific, albeit implicit, training choices.

Furthermore, the report explores the philosophical and linguistic dimensions of this capability, utilizing frameworks such as Leonard Talmy’s Force Dynamics and the theory of Implicit Causality (IC) verbs to benchmark LLM performance against human cognitive patterns. The evidence suggests that while LLMs have mastered the *linguistic interface* of causality—the "language game" of cause and effect -- significant questions remain regarding the grounding of these symbols in a genuine world model.

3. The Generative Era (2020–2025): Structural Induction of Causal Logic

The user's observation that models "since around ChatGPT 3.5" (released late 2022) exhibit a distinct causal proficiency aligns with the industry's shift toward **Instruction Tuning (IT)** and **Reinforcement Learning from Human Feedback (RLHF)**. The analysis of research data indicates that this proficiency is not a coincidence, but the result of specific training methodologies that inadvertently acted as a massive "causal curriculum."

3.1 The "Coincidence" of Pre-training: Implicit World Models

Before discussing specific training, one must acknowledge the foundation: pre-training on web-scale corpora (The Pile, Common Crawl, C4). The primary objective of these models is next-token prediction.

Theoretical research suggests that optimizing for prediction error on a diverse corpus forces the model to learn a compressed representation of the data generating process -- effectively, a "world model". Because human language is intrinsically causal (we tell stories of *why* things happen), a model trained to predict the next word in a narrative must implicitly model causal physics.

- *Example:* To predict the token "shattered" following the context "The vase fell off the shelf and...", the model must encode the causal relationship between *falling (gravity)* and *shattering (impact)*.

Recent theoretical work on **Semantic Characterization Theorems** argues that the latent space of these models evolves to map the topological structure of these semantic relationships. Thus, the "native" understanding is partially a coincidence of the data's nature: the model learns causality because causality is the statistical glue of human discourse.

3.2 The Instruction Tuning Hypothesis: Specific Training via Templates

The transition from "text completer" (GPT-3) to "helpful assistant" (ChatGPT) was mediated by **Instruction Tuning**. This process involves fine-tuning the model on datasets of (Instruction, Output) pairs. An analysis of major instruction datasets -- **FLAN**, **OIG**, and **Dolly** -- reveals that they are saturated with causal reasoning tasks.

3.2.1 The FLAN Collection: The Template Effect

The **FLAN (Finetuned Language Net)** project was instrumental in this development. Researchers took existing NLP datasets (including causal extraction datasets) and converted them into natural language templates.

- **The Mechanism:** A classification task from the *COPA (Choice of Plausible Alternatives)* dataset, which asks for the cause of an event, was transformed into prompts like: *"Here is a premise: The man broke his toe. What was the cause?"*

- **The Scale:** FLAN 2022 aggregated over 1,800 tasks. By training on millions of examples where the input is a scenario and the output is a causal explanation, the model explicitly learned the linguistic patterns of *identifying influence*.

- **Mixed Prompting:** Crucially, FLAN mixed **Chain-of-Thought (CoT)** templates (which require intermediate reasoning steps using "therefore," "because," "so") with standard prompts. This trained the model not just to guess the answer, but to *generate the causal logic* leading to it.

This contradicts the idea that the capability is purely coincidental. The models were specifically drilled on millions of "causal identification" exercises, disguised as instruction following.

3.2.2 Open Instruction Generalist (OIG) and Dolly

The **OIG** and **Dolly** datasets expanded this to open-domain interactions. These datasets contain thousands of "brainstorming" and "advice" prompts.

- *Data Evidence:* An entry from the OIG dataset reads: *": I'm having trouble finding a good job, what can I do to improve my chances? : One thing you could do is..."*.

- *Implication:* To answer this, the model must access a causal chain: *Action (revise resume)* -> *Effect (better chances)*. The prevalence of "how-to" and "why" questions in these datasets forces the model to organize its internal knowledge into causal structures (Means-End reasoning).

3.3 Reinforcement Learning from Human Feedback (RLHF): The Coherence Filter

The final layer of "specific training" is **RLHF**. In this phase, human annotators rank model outputs based on preference.

- **Preference for Logic:** Research indicates that humans have a strong bias for **causal coherence**. A narrative that flows logically (Cause A -> Effect B) is rated higher than one that is disjointed.
- **Length and Explanation Bias:** RLHF has been shown to induce a "length bias," where models produce longer, more detailed explanations to secure higher rewards. In the context of causality, this encourages the model to generate elaborate causal chains.
- **Sycophancy:** However, this training can also lead to "hallucinated causality." If a user asks a leading question implies a false causation (e.g., "Why does the moon cause earthquakes?"), an RLHF-aligned model might generate a plausible-sounding but scientifically incorrect causal explanation to satisfy the user's premise, prioritizing "helpfulness" over "truth".

Conclusion on Training vs. Coincidence: The capability is a hybrid. The *potential* to understand causality is a coincidence of pre-training scale (World Models), but the *ability to natively identify and articulate* it in response to a prompt is the result of specific Instruction Tuning and RLHF regimens that prioritize causal templates and coherent explanation.

---

4. Linguistic Frameworks: Analyzing "Ordinary" Causation

The user's query emphasizes the "native ordinary language concept of causation." To understand this, we must look beyond computer science to **Cognitive Linguistics**. Recent research has extensively benchmarked LLMs against human linguistic theories, particularly **Talmy's Force Dynamics** and **Implicit Causality (IC)**.

4.1 Force Dynamics: Agonists and Antagonists in Latent Space

Leonard Talmy's theory of **Force Dynamics** posits that human causal understanding is rooted in the interplay of forces: an **Agonist** (the entity with a tendency towards motion or rest) and an **Antagonist** (the opposing force).

- *Linguistic Patterns:* "The ball kept rolling despite the grass" (Agonist: Ball; Antagonist: Grass). "He let the book fall" (Removal of Antagonist).
- *LLM Evaluation:* Recent studies have tested LLMs on translating and explaining these force-dynamic constructions.

    - **Findings:** GPT-4 demonstrates a sophisticated grasp of these concepts. When translating "He let the greatcoat fall" into languages like Finnish or Croatian, the model correctly selects verbs that convey "cessation of impingement" (allowing) rather than "onset of causation" (pushing).
    - **Implication:** This suggests that LLMs have acquired a **schematic semantic structure** of causality. They do not merely predict words; they map the *roles* of entities in a physical interaction. However, this capability degrades in abstract social contexts. For example, in the sentence "Being at odds with her father made her uncomfortable," models sometimes misidentify the Agonist/Antagonist relationship, struggling to map "emotional force" as accurately as "physical force".

4.2 Implicit Causality (IC) Verbs

Another major area of inquiry is **Implicit Causality (IC)**, which refers to the bias native speakers have regarding *who* is the cause of an event based on the verb used.

- *NP1-Bias (Subject):* "John **upset** Mary." (Why? Because *John* is annoying).
- *NP2-Bias (Object):* "John **scolded** Mary." (Why? Because *Mary* did something wrong).

Benchmarking Results: Research comparing LLM continuations to human psycholinguistic data reveals a high degree of alignment.

- **Coreference:** When prompted with "John amazed Mary because...", LLMs overwhelmingly generate continuations referring to John, matching human NP1 bias.
- **Coherence:** Humans tend to provide *explanations* following these verbs. LLMs mirror this "explanation bias," prioritizing causal connectives over temporal or elaborative ones in these contexts.
- **Significance:** This indicates that LLMs have encoded the **pragmatics of blame and credit** inherent in ordinary language. They "know" that "apologizing" implies the subject caused a negative event, while "thanking" implies the object caused a positive one. This is crucial for the "native" feel of their interactions—they navigate the social logic of causality fluently.

4.3 The Limits of "Native" Understanding: The Causal Parrot Debate

Despite these successes, a vigorous debate persists regarding whether this constitutes "understanding" or merely "stochastic parroting".

- **The "Parrot" Argument:** Critics argue that LLMs fail when the linguistic surface form is stripped away. On benchmarks like **CausalProbe** , which uses fresh, non-memorized data, model performance drops significantly. This suggests that LLMs rely on **Level 1 (Association)** reasoning—pattern matching seen examples—rather than **Level 2 (Intervention)** reasoning.

- **The "Simulacrum" Argument:** Conversely, the **Semantic Characterization Theorem** proposes that the model's high-dimensional space creates a functional topology that is mathematically equivalent to a discrete symbolic system. Even if the model has never "seen" a glass break, its representation of "glass" and "break" are topologically linked in a way that allows it to simulate the causal outcome efficiently.

---

5. Benchmarking the "Informal": From Social Media to Counterfactuals

The evaluation of causal understanding has evolved from F1 scores on extraction tasks to sophisticated benchmarks that test the model's ability to handle the messy, informal causality of the real world.

5.1 CausalTalk: Informal Causality in Social Media

The **CausalTalk** dataset addresses the user's interest in "passages where one thing influences another" in informal contexts.

- *The Challenge:* In social media (e.g., Reddit), causality is often expressed without explicit markers. "I took the vaccine and now I feel sick" contains no "because," yet the causal assertion is clear.
- *Findings:* LLMs show remarkable proficiency in identifying these **implicit causal claims**, often outperforming traditional supervised models. They can detect "gist" causality—the overall causal assertion of a post—even when it is buried in sarcasm or non-standard grammar.
- *Application:* This is critical for **misinformation detection**. Models are being used to identify exaggerated causal claims in science news (e.g., reporting a correlation as a causation). However, LLMs sometimes struggle to distinguish between a user *reporting* a correlation and *asserting* a causation, highlighting a nuance gap in their "native" understanding.

5.2 Explicit vs. Temporal Confusion (ExpliCa)

The **ExpliCa** benchmark investigates a specific failure mode: the confusion of time and cause.

- *The Fallacy: Post hoc ergo propter hoc* ("After this, therefore because of this").
- *LLM Behavior:* Research shows that LLMs are prone to this fallacy. When events are presented in chronological order ("The sun set. The streetlights turned on."), models are statistically more likely to infer a causal link than humans, who might see it as mere sequence. This suggests that the "native" understanding is heavily biased by the **narrative structure** of training data, where chronological sequencing often implies causality.

5.3 Counterfactuals and "What If" (CRASS)

The **CRASS** (Counterfactual Reasoning Assessment) benchmark tests the model's ability to reason about what *didn't* happen.

- *Task:* "A man drinks poison. What would have happened if he drank water?"
- *Results:* While base models perform adequately, fine-tuning with techniques like **LoRA (Low-Rank Adaptation)** significantly boosts performance. This reinforces the "training hypothesis"—the capacity for causal reasoning is latent in the weights but requires specific activation (instruction tuning) to be robustly deployed.

---

6. Philosophical Dimensions: Symbol Grounding and World Models

The impressive performance of LLMs on causal tasks raises profound philosophical questions about the nature of meaning. Can a system that has never physically interacted with the world truly understand "force," "push," or "cause"?

6.1 The Symbol Grounding Problem

Cognitive scientists have long argued that human concepts are **grounded** in sensorimotor experience. We understand "heavy" because we have felt gravity.

- **The Disembodied Mind:** LLMs are disembodied. Their understanding of "force" is purely distributional—"force" is defined by its mathematical proximity to "push," "move," and "impact" in vector space.
- **Cognitive Alignment:** Research using the **Brain-Based Componential Semantic Representation (BBSR)** shows that LLM representations align well with human cognition for concrete concepts but diverge for embodied experiences (e.g., olfaction, gustation) and spatial cognition.
- **Functional Understanding:** However, proponents of the "Functionalist" view argue that if an LLM can answer "What happens if I drop this?" indistinguishably from a human, it possesses a **functional understanding** of causality. The **Semantic Characterization Theorem** supports this by demonstrating that continuous learning dynamics can give rise to stable, discrete semantic attractors that behave like symbolic rules.

6.2 Causal Determinism vs. Autoregressive Generation

A critical distinction exists between traditional causal inference (which assumes a stable structural model) and LLM generation (which is probabilistic and autoregressive).

- *Drift:* An LLM generates a causal explanation token-by-token. Research indicates that this can lead to **causal drift**, where the model "changes its mind" mid-sentence if the probability distribution shifts.

- *Hallucination:* This is the root of causal hallucination. The model is not traversing a logical graph; it is surfing a wave of probability. If the most likely next word contradicts the causal logic of the previous ten words, the model may output it anyway, sacrificing causal consistency for local fluency.

7. Current Frontiers (2024–2025): Reasoning Models and Future Directions

The field is currently undergoing another shift with the introduction of "Reasoning Models" (e.g., OpenAI's o1/o3 series, DeepSeek R1).

7.1 Chain-of-Thought Monitoring and "Thinking" Tokens

Newer models are trained to produce hidden "chains of thought" before generating a final answer.

- *Impact on Causality:* This allows the model to perform **intermediate causal checks**. Instead of predicting the effect immediately, the model can "reason" silently: *Premise* -> *Mechanism* -> *Potential Confounders* -> *Conclusion*.
- *Research Findings:* Snippet discusses "CoT Monitoring," showing that these internal reasoning traces can be monitored to detect "reward hacking" or deceptive alignment. This suggests a move toward making the model's implicit causal reasoning **explicit** and **verifiable**.

7.2 Causal Graph Construction

Recent work has moved back to structure, using LLMs to *extract* and *construct* **Causal Graphs** (DAGs) from unstructured text.

- *Method:* Rather than asking the LLM to just "answer," researchers prompt it to output a graph: `Nodes:, Edges:`.
- *Result:* This leverages the LLM's linguistic fluency to structure knowledge, which can then be processed by formal causal inference algorithms, bridging the gap between "informal ordinary language" and "formal causal calculus."

8. Conclusion

The research of the last decade confirms that the "native" causal understanding of LLMs is a constructed capability, forged in the fires of massive data and refined by human-centric training. It is not a coincidence, but a predictable outcome of optimizing models to predict a world that is inherently causal.

1. **Origin:** The capability originates in **pre-training**, where the model learns the distributional "shadow" of causation cast by billions of human sentences.

2. **Development:** It is sharpened by **Instruction Tuning** (FLAN, Dolly), which explicitly teaches the model the "language game" of explanation and consequence through millions of templates.
3. **Refinement:** It is polished by **RLHF**, which imposes a human preference for logical coherence and narrative flow, effectively pruning non-causal outputs.
4. **Nature:** This understanding is **linguistic and schematic**. It mirrors the force dynamics and implicit biases of human language with uncanny accuracy but remains brittle when faced with novel physical interactions or rigorous counterfactual logic.

For the user impressed by this ability: You are witnessing a system that has learned to simulate the *reasoning patterns* of humanity. It understands "cause" not as a physical law, but as a linguistic necessity—a rule of grammar for the story of the world.

9. Comparative Data Tables

Table 1: Evolution of Causal Tasks and Metrics (2015–2025)

Era	Primary Focus	Methodology	Dominant Datasets	Typical Metric	"Native" Capability
2015–2018	Relation Classification	SVM, RNN, Sieves	SemEval-2010 Task 8, EventStoryLine	F1 Score (~0.50-0.60)	None (Pattern Matching)
2019–2021	Span/Context Extraction	BERT, RoBERTa	Causal-TimeBank, BioCausal	F1 Score (~0.72)	Contextual Recognition
2022–2025	Generative Reasoning	GPT-4, Llama, Instruction Tuning	CausalTalk, CRASS, ExpliCa	Accuracy, Human Eval	Generative/Schematic

Table 2: Performance on Causal Benchmarks (Selected Studies)

Benchmark	Task Description	Model Class	Performance Note	Source
SemEval Task 8	Classify relation between nominals	BERT-based (BioBERT)	~0.72-0.80 F1 (High accuracy on explicit triggers)	
CRASS	Counterfactual "What if" reasoning	GPT-3.5 / Llama	Moderate baseline; significantly improved with LoRA/PEFT	
CausalProbe	Causal relations in *fresh* (unseen) text	GPT-4 / Claude	Significant drop compared to training data; suggests memorization	
Implicit Causality	Predicting subject/object bias (John amazed Mary)	GPT-4	High alignment with human psycholinguistic baselines	
Force Dynamics	Translating "letting/hindering" verbs	GPT-4	High accuracy in preserving agonist/antagonist roles	

Table 3: Key Instruction Tuning Datasets Influencing Causal Capability

Dataset	Content Type	Causal Relevance	Mechanism of Training	Source
FLAN	NLP Tasks -> Instructions	High (COPA, e-SNLI templates)	Explicitly maps "Premise" -> "Cause/Effect" in mixed prompts	
OIG	Open Generalist Dialogues	High (Advice, How-to)	Teaches Means-End reasoning (Action -> Result)	
Dolly	Human-generated Q&A	High (Brainstorming, QA)	Reinforces human-like explanatory structures	
CausalTalk	Social Media Claims	High (Implicit assertions)	Captures "gist" causality in informal discourse	

A formalisation of causal mapping

Abstract

Draft for an IJSRM submission. This paper proposes a lightweight grammar and logic for encoding, aggregating, and querying causal claims found in qualitative text data.

The specification is grounded in a "Minimalist" (or "Barefoot") approach to causal coding: it prioritises capturing the explicit causal claims made by sources, without imposing complex theoretical frameworks that may not align with how people naturally speak.

1. Data Structures

The foundation of the specification is the **Project Data** package, which strictly separates the causal claims from the source material.

Definition Rule DS-PROJECTDATA: Project Data

- **Definition:** `ProjectData = LinksTable + optional SourcesTable`
- **Note:** `SourcesTable` may be omitted (or empty). If it is present, the evidence constraint in DS-LINKS-SOURCES applies.

Syntax Rule DS-LINKS: The Basic Links Table

A list of causal connections where each row represents one atomic claim.

- **Structure:** A table containing at least the following columns:
- `Cause`: Text label for the driver (influence factor).
- `Effect`: Text label for the outcome (consequence factor).
- `Link_ID`: Unique identifier for bookkeeping.
- `Context`: Optional identifier for distinguishing contexts.
- `Source_ID`: Identifier for the origin of the claim (e.g., document ID, participant ID).
- **Extensions:** The table may contain additional columns (e.g., `Sentiment`, `Time_Period`) or tags.

Syntax Rule DS-SOURCES: The Sources Table (optional)

A registry of documents, interviews, or respondents.

- **Structure:** A table containing:
- `Source_ID`: Unique key.
- `Full_Text`: The complete text of the source document.
- `Metadata`: Optional custom columns representing attributes of the source (e.g., Gender, Region, Date).

Syntax Rule DS-LINKS-SOURCES: Evidence Constraint (if Sources table is present)

- **Additional column:** The Links table also contains:

- `Quote`: The specific text segment evidencing the claim.
- **Constraint:** If a Sources table is provided, then for every link `L` in the Links table:
- `L.Source_ID` MUST match a `Sources.Source_ID`, and
- `L.Quote` MUST be a substring of `Sources.Full_Text` for the matching `Source_ID`.

Coding is strictly evidence-based.

Definition Rule DS-FACTORS: The Factors Derivation

There is no independent table for Factors stored in the Project. Factors are derived entities.

- **Definition:** `Factors = unique values in LinksTable.Cause + LinksTable.Effect`
- **Note:** A "Factors Table" is a transient structure created only during analysis.

2. Coding and Semantics

This section defines how text is translated into the data structures defined above.

Definition Rule COD-DEF: The Definition of Coding

Coding is the process of extracting links from source text into the Links table.

Semantics Rule COD-ATOM: The Atomic Causal Claim

A single row in the Links table, `Link | Cause="A" | Effect="B" | Source_ID="S1"`, is interpreted semantically as:

- *"Source S1 claims that A causally influenced B in virtue of its causal power to do so."*

Semantics Rule COD-BARE: Bare Causation

The relationship `A -> B` implies **influence**, not determination.

- **Negative Definition:** It does NOT imply `A` is the only cause of `B`.
- **Negative Definition:** It does NOT imply `A` is sufficient for `B`.

Example:

- **Source Text:** "Because of the drought, we had to sell our livestock." (from Source 12)
- **Coded Link:**
- Cause: `Drought`
- Effect: `Selling livestock`
- Source_ID: `12`
- Quote: "Because of the drought, we had to sell our livestock"

Semantics Rule COD-MIN: Minimalist Coding Principles

The coding schema prioritizes explicit claims over complex theoretical frameworks.

- **The 90% Rule:** Code the simple form **"X causally influenced Y"**. Explicit coding of complex logic (enablers, sufficiency, non-linearity) is not provided for.

- **Propositions, Not Variables:** Factors are simple propositions (e.g., "Jo started shouting"), not variables with values. Distinct concepts should not be merged prematurely.
- **Example (why this matters):** Code `Poverty` and `Wealth` as distinct factors rather than values of one "Economic Status" variable. A source may claim `Poverty -> Stress` and also `Wealth -> Investment`; collapsing these too early can obscure distinct narratives.
- **No Absences:** Do not code absences. If a source does not mention a factor, it is unknown, not absent.
- **Realism & Partiality:** "X influenced Y" means X had the causal power to affect Y in this context. It implies a contribution, not total determination.

Surface cognition

- **Goal:** Model the speaker's expressed causal thinking (cognition), not necessarily underlying objective reality.
- **Method:** Code only what is explicitly said; avoid inferring hidden variables or unstated counterfactuals.

Semantics Rule COD-MULTI: The meaning of multiple links

A table containing multiple links simply asserts the logical conjunction of the links:

- Source S1 claims that A causally influenced B in virtue of its causal power to do so.
- Source S1 claims that C causally influenced D in virtue of its causal power to do so.
- Source S2 claims that A causally influenced C in virtue of its causal power to do so.

So, unless specific contexts are specified, if Source S1 claims that `A -> C` and also that `B -> C`, this is neither a contradiction nor (by default) a claim that A-and-B jointly influenced C as a package. It is simply two separate claims.

3. The Filter Pipeline (Query Language)

Analysis is performed by passing a links table through a sequence of filters.

Syntax Rule FIL-PIPE: The Semantic Pipeline

The meaning of a result is defined by the cumulative semantic restrictions or transformations of the filters applied.

- **Syntax:** `Input |> Filter1 |> Filter2 |> Output`
- **Multi-line Syntax:**

```
Input
    |> Filter1
    |> Filter2
    |> Output
```

- **Processing:** Filters are applied sequentially. The output of `Filter N` becomes the input of `Filter N+1`.

Types of Filters

- **All filters take a Links table as input.** Most filters return a Links table (so they can be chained). **Output filters terminate the pipeline** by returning a derived view (table/summary) rather than a Links table.

In practice (as in the app), filter behaviour is often **multi-effect**. For example, a filter may rewrite labels *and* add tracking columns. So instead of forcing each filter into exactly one “type”, we treat each filter as having an **effect signature**:

- **Row selection**: changes *which links* are included (drops/retains rows).
- **Label rewrite**: changes `Cause`/`Effect` labels (recoding/normalisation).
- **Column enrichment**: adds or recalculates columns (metadata/metrics/flags), typically to support later filtering or display.
- **Configuration**: changes display/formatting settings without changing the links table (app-specific; not formalised here as a core links-table transform).

Below, filters are grouped by their **primary intent**, and each rule declares its **Effects:** line.

Row-selection filters

Syntax Rule FIL-CTX: Context Filters

Reduces the evidentiary base based on Source metadata.

- **Effects:** row selection
- **Operation:** `filter_sources | <criteria...>`
- **Semantics:** Restrict the evidence base to links whose `Source_ID` is in the retained set of sources.

Syntax Rule FIL-FREQUENCY: Content Filters

Reduces the evidentiary base based on signal strength.

- **Effects:** row selection
- **Operation:** `filter_links | <criteria...>`
- **Semantics:** Retain only links meeting an evidence threshold (e.g., `min_source_count=2`).

Syntax Rule FIL-TOPO: Topological Filters

Retains links based on their position in a causal chain.

- **Effects:** row selection
- **Operation:** `trace_paths | from="<factor>" | to="<factor>" | <options...>`
- **Semantics:** "Retaining only mechanisms that connect *From* to *To*."

Label-rewrite filters

Semantics Rule FIL-ZOOM: The Zoom Filter (Hierarchical Syntax)

Extends the logic to handle nested concepts via a separator syntax.

- **Effects:** label rewrite
- **Syntax:** Factors may use the `;` separator (e.g., `General Concept; Specific Concept`).
- **Semantics:** `A; B` implies `B` is an instance or sub-component of `A`.
- **Operation:** `transform_labels | zoom_level=1`. If `zoom_level=1`, rewrite labels by truncating text after the first separator.
- **Inference:** At Zoom Level 1, `A; B` is treated logically as `A`.

Semantics Rule FIL-OPP: The Combine Opposites Filter (Bivalence Syntax)

Extends the logic to handle polarity/negation.

- **Effects:** label rewrite; column enrichment
- **Syntax:** Factors may use the `~` prefix (e.g., `~Employment`) or tag pairs.
- **Semantics:** `~A` is the negation of `A`.
- **Operation:** `combine_opposites`. Rewrites negative labels (e.g., `~Income`) to their positive counterparts (`Income`) and adds tracking columns such as `flipped_cause` and `flipped_effect`.
- **Inference:** Evidence for `~A -> ~B` is treated as corroborating evidence for `A -> B` (with flipped polarity).

Column-enrichment filters

Syntax Rule FIL-BUNDLE: The Bundling Filter

This filter aggregates co-terminal links (links with the same cause and effect) to calculate evidence metrics without reducing the row count. We normally think of it as being automatically applied after any other filter.

- **Effects:** column enrichment
- **Operation:** `bundle_links`
- **Definition (bundle object):** `Bundle(A, B) = all links L where L.Cause <mark> A AND L.Effect </mark> B`
- **Logic:** For every link `L`, identify the set of all links `S` where `S.Cause <mark> L.Cause` AND `S.Effect </mark> L.Effect`.
- **Transformation:** Appends new columns to the Links table:
- `Bundle`: For convenience, a text representation of the connection (e.g., "A -> B").
- `Citation_Count`: The total count of rows in set `S`. Represents volume of coding.
- `Source_Count`: The number of unique `Source_IDs` in set `S`. Represents breadth of evidence (consensus).
- **Lemma:** `Source_Count <= Citation_Count`.

We measure importance using two distinct metrics:

- **Citation Count:** The total number of times a link or factor was mentioned across the entire project. This counts every single row in the data.
- *Technical:* `citation_count`
- **Source Count (or Number of People):** The number of *unique* sources (people or documents) that mentioned a link or factor. This avoids double-counting if one person repeats the same point multiple times.
- *Technical:* `source_count`

Output filters

Output Rule OUT-FACTORS: Factors table view

Returns a Factors table (one row per factor) derived from a Links table (typically after `FIL-BUNDLE`).

- **Operation:** `factors_view`
- **Semantics:** Aggregate over the set of factor labels appearing anywhere in `Cause` or `Effect`, and compute per-factor summaries (e.g., role metrics).

Output Rule OUT-MAP: Graphical map view

Returns a graphical network view of the current Links table.

- **Operation:** `map_view`
- **Semantics (data):**
- **Nodes:** Factors (labels appearing in `Cause` or `Effect`).
- **Edges: Bundles** (one edge per `Cause -> Effect` pair), built from the **current filtered/transformed labels** (so the map reflects Zoom/Combine-Opposites/etc.).
- **Bundling:** If bundle-level columns are not already present, the map view implicitly applies `FIL-BUNDLE` to compute bundle metrics (e.g., `Citation_Count`, `Source_Count`) used for labels and styling.
- **Semantics (presentation):** The view includes a formatting configuration that maps derived metrics to visual encodings (e.g., edge width by citation/source count; edge colour/arrowheads by mean sentiment; node colour by outcomeness; node/label sizes by frequency), plus a legend summarising the current view and applied filters.

Definition Rule MET-NODE: Factor Role Metrics

These metrics describe the topological role of a factor.

- **In-Degree (incoming citations):** Count of incoming links (times this factor appears as an Effect).
- *Technical:* `citation_count_in`
- **Out-Degree (outgoing citations):** Count of outgoing links (times this factor appears as a Cause).
- *Technical:* `citation_count_out`
- **Outcomeness:** `In-Degree / (In-Degree + Out-Degree)`.

- *Interpretation:* A score nearing 1 indicates an Outcome; a score nearing 0 indicates a Driver.

4. Example Queries

Example A: The "Drivers" Query *Question: What do female participants say are the main drivers of Income?*

```
Result = ProjectData
  |> filter_sources | Gender="Female"         // Rule FIL-CTX
  |> trace_paths | to="Income" | steps=1      // Rule FIL-TOPO
  |> filter_links | min_citations=2           // Rule FIL-FREQUENCY
```

Example B: The "Mechanism" Query *Question: Is there valid narrative evidence that Training leads to Better Yields?*

```
Result = ProjectData
  |> transform_labels | zoom_level=1                   // Rule FIL-ZOOM
  |> trace_paths | from="Training" | to="Yield" | thread_tracing=TRUE   // Rule FIL-TOPO
```

5 Causal Inference?

Inference Rule INF-EVID: Evidence is not effect size

We quantify **evidence strength**, not **causal effect strength**.

- **Observation:** `Link | Cause="A" | Effect="B"` appears 10 times.
- **Inference:** There are 10 pieces of evidence (10 coded mentions) for the claim `A -> B`.
- **Invalid inference:** The influence of A on B is 10 times stronger than a link appearing once.

Inference Rule INF-FACT: Factual Implication

If we observe `Link | Cause="A" | Effect="B" | Source_ID="S1"`:

- **Deduction:** `Source S1 claims A happened/exists`.
- **Deduction:** `Source S1 claims B happened/exists`.

Inference Rule INF-THREAD: Thread Tracing (Valid Transitivity)

We can infer a long causal chain (indirect influence) only if one source provides every step.

- **Logic:** `Link | Cause="A" | Effect="B" | Source_ID="S1"` AND `Link | Cause="B" | Effect="C" | Source_ID="S1"` => Valid path `A -> B -> C`.

Inference Rule INF-CTX: The Context Rule (The Transitivity Trap)

We cannot infer causal chains by stitching together different sources without checking context.

- **Logic:** `Link | Cause="A" | Effect="B" | Source_ID="S1"` AND `Link | Cause="B" | Effect="C" | Source_ID="S2"` `=>` INVALID path `A -> C`, unless S1 and S2 share the same `Context`.

Appendix A: AI Extensions

These filters extend the core logic using probabilistic AI models (Embeddings or Clustering).

Semantics Rule FIL-SOFT: The Soft Recode Filter

Extends logic using semantic similarity (vector embeddings).

- **Operation:** `soft_recode | magnets="<...>" | similarity_threshold="<...>"`
- **Inference:** If `Label A` is similar to `Magnet M` (> threshold), treat `A` as `M`.

Semantics Rule FIL-AUTO: The Auto Recode Filter

Extends logic using unsupervised clustering.

- **Operation:** `auto_recode | target_clusters=K`
- **Inference:** Factors group together based on inherent semantic proximity into `K` emergent themes.