Can AI Text Watermark Detection Prove a Document Was Machine-Written?

Maria Cattini | 20/08/2026 | AI

A statement arrives at 23:40, three hours before your story runs. It is fluent, well-structured, and slightly too clean. You want one thing: to know whether a person wrote it or a model did. Since 2 August 2026, Article 50 of the EU AI Act applies, and providers of systems that generate synthetic text must mark their outputs in a machine-readable way and make them detectable as artificially generated. So a reasonable analyst asks a reasonable question: can I now just run the file through a checker?

The short answer is that **AI text watermark detectio**n gives you a usable positive signal and almost no negative one. In a July 2026 post, engineer Sean Goedecke argued that text watermarks will always be trivial to remove. His technical reasoning holds up, and it changes how the check should sit inside an OSINT workflow: as one weak indicator, never as proof.

Why is text so much harder to watermark than an image?

An image carries noise the eye ignores. You can force twenty pixels to share a colour and nobody notices. Text has no such slack. It is a compressed medium: change a sentence and a reader sees the change. That is why watermarking text is a steganography problem — hiding a signal inside a message — with the extra constraint that the message itself cannot be freely manipulated. Goedecke's toy example makes the limit obvious: a rule like "every fifth letter is an 'e'" would be a perfect fingerprint and would fill the output with typos.

There is an intuitive alternative that fails for a different reason. If a lab can run a text back through its own models and measure how closely the predicted tokens match, why watermark at all? Because the set of answers a model could produce is far larger than the set of watermarked answers, so you drown in false positives from humans who happen to write like a chatbot. Running every model against every submitted text is also expensive — and the compliance direction points towards free, public detection tools rather than costly per-query inference.

How do the current watermarking methods actually work?

Two families are in play, and they fail in different ways.

The first is statistical, and **Google's SynthID** is the reference case — the only text watermarking a provider has publicly described. A model does not emit one token at a time; it emits a probability distribution over its whole vocabulary and samples from the top candidates. SynthID assigns each candidate a score derived from the preceding tokens, then biases sampling toward high-scoring tokens. Detection means re-scoring a block of text and asking whether the aggregate is improbably high. It is the em-dash folklore made rigorous: instead of a list of suspicious words, a mathematical relationship between words that no reader can see, and that costs almost nothing to check.

The second is typographic. A normal space (U+0020) can be swapped for a three-per-em space (U+2004) or a CJK ideographic space (U+3000). These near-identical characters are called homoglyphs, and a pattern of them — say, every third space — is rare enough in the wild to work as

a fingerprint. It is cheaper than SynthID and can be applied client-side, after generation, without touching the model's choices.

Keep the evidential status of that second family straight, because it is where speculation creeps in. Goedecke documents that homoglyphs have been used by AI products: Claude Code used them to tag suspicious requests, a behaviour later rolled back, and pasted ChatGPT output has shown up in code editors with unusual space characters, which OpenAI attributed to a model quirk. That homoglyphs appear is documented. That they are being used specifically as compliance watermarks is his hypothesis, and he says so. Treat it as a hypothesis in your own notes.

Signed metadata does not close the gap either. C2PA-style content credentials attach a signature to a file, so a container can be trusted or held in suspicion. Chat output is not a container. Plain text pasted into an email has nothing to sign.

What should an analyst do with a suspect text?

The point of the workflow below is not to declare "AI" or "human". It is to extract what the artefact can actually support, and to record what it cannot.

1. Preserve the original bytes before touching anything. Save the raw file or the raw paste, hash it, and work on a copy. Any cleanup you do — a paste through a plain-text editor, an autocorrect pass — can destroy exactly the character-level evidence you are looking for.
2. Inspect the copy for unusual Unicode. Open it in an editor that reveals invisible or non-standard characters, or view the byte values directly. You are looking for spaces, apostrophes and dashes that are not the standard ones, and for whether the anomalies form a pattern rather than appearing at random.
3. Record the pattern, not the verdict. Note which characters appear, how often, and where. Human text picks up homoglyphs from word processors, CMS editors and copy-paste chains all the time. A regular interval is more interesting than a single oddity.
4. Run any provider watermark check that exists for the suspected source, and log the result as one indicator. A positive result is meaningful. A negative result tells you very little, because a single paraphrase through any small model rewrites the vocabulary choices the statistical watermark depends on, and a find-and-replace strips the typographic one. Someone with access to a public checker can iterate until the text comes back clean.
5. Shift weight to non-linguistic evidence. Document provenance, sending infrastructure, publication history, timestamps, the account that distributed the file, and whether the sender confirms authorship. These do not degrade when the text is reworded.
6. Write the uncertainty into the output. Separate what is verified (this file contains these characters), what is inferred (the distribution looks deliberate), and what is unverified (which system produced it, if any).

The recurring mistakes are predictable. Analysts treat a negative check as evidence of human authorship. They report a detector's percentage as if it were a measurement. They accuse a named author on stylistic grounds — the fastest route to a correction. And they let the origin question swallow the one that usually matters more: whether the claims in the document are true.

A concrete illustration, offered as a scenario rather than a case: a small association receives a supplier letter announcing new payment details. The letter passes a watermark check cleanly. Under step 2 the spaces turn out to be uniform and ordinary — consistent with human typing, and equally consistent with a paraphrase pass. The origin question stalls there, which is the correct outcome. What resolves the situation is step 5: the domain registration date, the mail headers, and a phone call to a number the association already had on file.

The expected result of the workflow is not certainty. It is a document that says clearly which parts of the origin question are answered and which are open — and that survives being challenged.

Why does this matter beyond compliance?

For providers, the regulatory pressure is real: Article 50 carries fines up to €15 million or 3% of worldwide turnover, and systems already on the market have until 2 December 2026 for the marking obligation. For everyone downstream, the practical consequence runs the other way. Marking will make some AI content easier to identify, which raises the perceived authority of a clean check — precisely the inference the technology cannot support. Newsrooms that build "we verified it was not AI-generated" into their process are buying a claim they cannot defend. Small organisations relying on a checker to filter fraudulent correspondence are protecting themselves against the careless and not against anyone who has read a removal guide.

The operational criterion is narrow enough to remember. A watermark hit is a lead worth pursuing. A watermark miss is not a finding, and should never appear in your report as one. Origin questions about plain text get resolved by provenance, infrastructure and confirmation from the source — the same evidence you would need if watermarking had never been mandated at all.

A statement arrives at 23:40, three hours before your story runs. It is fluent, well-structured, and slightly too clean. You want one thing: to know whether a person wrote it or a model did. Since 2 August 2026, Article 50 of the EU AI Act applies, and providers of systems that generate synthetic text must mark their outputs in a machine-readable way and make them detectable as artificially generated. So a reasonable analyst asks a reasonable question: can I now just run the file through a checker?

The short answer is that **AI text watermark detectio**n gives you a usable positive signal and almost no negative one. In a July 2026 post, engineer Sean Goedecke argued that text watermarks will always be trivial to remove. His technical reasoning holds up, and it changes how the check should sit inside an OSINT workflow: as one weak indicator, never as proof.

Why is text so much harder to watermark than an image?

An image carries noise the eye ignores. You can force twenty pixels to share a colour and nobody notices. Text has no such slack. It is a compressed medium: change a sentence and a reader sees the change. That is why watermarking text is a steganography problem — hiding a signal inside a message — with the extra constraint that the message itself cannot be freely manipulated. Goedecke's toy example makes the limit obvious: a rule like "every fifth letter is an 'e'" would be a perfect fingerprint and would fill the output with typos.

There is an intuitive alternative that fails for a different reason. If a lab can run a text back through its own models and measure how closely the predicted tokens match, why watermark at all? Because the set of answers a model could produce is far larger than the set of watermarked answers, so you drown in false positives from humans who happen to write like a chatbot. Running every model against every submitted text is also expensive — and the compliance direction points towards free, public detection tools rather than costly per-query inference.

How do the current watermarking methods actually work?

Two families are in play, and they fail in different ways.

The first is statistical, and **Google's SynthID** is the reference case — the only text watermarking a provider has publicly described. A model does not emit one token at a time; it emits a probability distribution over its whole vocabulary and samples from the top candidates. SynthID assigns each candidate a score derived from the preceding tokens, then biases sampling toward high-scoring tokens. Detection means re-scoring a block of text and asking whether the aggregate is improbably high. It is the em-dash folklore made rigorous: instead of a list of suspicious words, a mathematical relationship between words that no reader can see, and that costs almost nothing to check.

The second is typographic. A normal space (U+0020) can be swapped for a three-per-em space (U+2004) or a CJK ideographic space (U+3000). These near-identical characters are called homoglyphs, and a pattern of them — say, every third space — is rare enough in the wild to work as a fingerprint. It is cheaper than SynthID and can be applied client-side, after generation, without

touching the model's choices.

Keep the evidential status of that second family straight, because it is where speculation creeps in. Goedecke documents that homoglyphs have been used by AI products: Claude Code used them to tag suspicious requests, a behaviour later rolled back, and pasted ChatGPT output has shown up in code editors with unusual space characters, which OpenAI attributed to a model quirk. That homoglyphs appear is documented. That they are being used specifically as compliance watermarks is his hypothesis, and he says so. Treat it as a hypothesis in your own notes.

Signed metadata does not close the gap either. C2PA-style content credentials attach a signature to a file, so a container can be trusted or held in suspicion. Chat output is not a container. Plain text pasted into an email has nothing to sign.

What should an analyst do with a suspect text?

The point of the workflow below is not to declare "AI" or "human". It is to extract what the artefact can actually support, and to record what it cannot.

1. Preserve the original bytes before touching anything. Save the raw file or the raw paste, hash it, and work on a copy. Any cleanup you do — a paste through a plain-text editor, an autocorrect pass — can destroy exactly the character-level evidence you are looking for.
2. Inspect the copy for unusual Unicode. Open it in an editor that reveals invisible or non-standard characters, or view the byte values directly. You are looking for spaces, apostrophes and dashes that are not the standard ones, and for whether the anomalies form a pattern rather than appearing at random.
3. Record the pattern, not the verdict. Note which characters appear, how often, and where. Human text picks up homoglyphs from word processors, CMS editors and copy-paste chains all the time. A regular interval is more interesting than a single oddity.
4. Run any provider watermark check that exists for the suspected source, and log the result as one indicator. A positive result is meaningful. A negative result tells you very little, because a single paraphrase through any small model rewrites the vocabulary choices the statistical watermark depends on, and a find-and-replace strips the typographic one. Someone with access to a public checker can iterate until the text comes back clean.
5. Shift weight to non-linguistic evidence. Document provenance, sending infrastructure, publication history, timestamps, the account that distributed the file, and whether the sender confirms authorship. These do not degrade when the text is reworded.
6. Write the uncertainty into the output. Separate what is verified (this file contains these characters), what is inferred (the distribution looks deliberate), and what is unverified (which system produced it, if any).

The recurring mistakes are predictable. Analysts treat a negative check as evidence of human authorship. They report a detector's percentage as if it were a measurement. They accuse a named author on stylistic grounds — the fastest route to a correction. And they let the origin question swallow the one that usually matters more: whether the claims in the document are true.

A concrete illustration, offered as a scenario rather than a case: a small association receives a supplier letter announcing new payment details. The letter passes a watermark check cleanly. Under step 2 the spaces turn out to be uniform and ordinary — consistent with human typing, and equally consistent with a paraphrase pass. The origin question stalls there, which is the correct outcome. What resolves the situation is step 5: the domain registration date, the mail headers, and a phone call to a number the association already had on file.

The expected result of the workflow is not certainty. It is a document that says clearly which parts of the origin question are answered and which are open — and that survives being challenged.

Why does this matter beyond compliance?

For providers, the regulatory pressure is real: Article 50 carries fines up to €15 million or 3% of

worldwide turnover, and systems already on the market have until 2 December 2026 for the marking obligation. For everyone downstream, the practical consequence runs the other way. Marking will make some AI content easier to identify, which raises the perceived authority of a clean check — precisely the inference the technology cannot support. Newsrooms that build "we verified it was not AI-generated" into their process are buying a claim they cannot defend. Small organisations relying on a checker to filter fraudulent correspondence are protecting themselves against the careless and not against anyone who has read a removal guide.

The operational criterion is narrow enough to remember. A watermark hit is a lead worth pursuing. A watermark miss is not a finding, and should never appear in your report as one. Origin questions about plain text get resolved by provenance, infrastructure and confirmation from the source — the same evidence you would need if watermarking had never been mandated at all.