Un agente AI che non si limita a cercare: verifica quello che trova

Maria Cattini | 11/08/2026 | Intelligenza Artificiale

Chi fa due diligence su una persona o un'azienda conosce bene il problema centrale: non manca mai l'informazione, manca la certezza che quell'informazione sia coerente. Un ruolo professionale che su un sito risulta attivo dal 2017 e su un altro si interrompe nel 2019. Una relazione tra due enti che sembra diretta ma nasce solo da una coincidenza di date. Il lavoro di verifica open source è soprattutto questo: incrociare, contraddire, scartare.

Un progetto open source chiamato **Deep Research AI Agent** (repository ai-assessment su GitHub, sviluppato da Harsh Solanki) prova a rendere replicabile proprio questa fase, quella più lenta e più spesso saltata. Non è un prodotto commerciale: è nato come esercizio per una selezione tecnica (Deriv AI, AI Engineer Technical Assessment), pubblicato con licenza MIT. Ma il modo in cui è costruito vale la pena guardarlo da vicino, perché il metodo che applica — pianificazione separata dall'esecuzione, costo modulato sulla qualità richiesta, fatti ancorati nel tempo, relazioni rappresentate come grafo — non riguarda solo questo strumento specifico.

Come è organizzato il lavoro

Il sistema usa **LangGraph**, un framework che permette di costruire flussi di lavoro con cicli e decisioni condizionali, non solo sequenze lineari. Al centro c'è un nodo che il progetto chiama Research Director: pianifica il passo successivo, decide quale agente specializzato attivare — ricerca web, estrazione di fatti, analisi temporale, risoluzione delle entità duplicate, analisi del rischio, mappatura delle connessioni, verifica delle fonti, generazione del report — e valuta quando fermarsi. In totale, contando anche il Direttore, sono nove nodi coordinati. La ricerca vera e propria attraversa sei fasi in sequenza: raccolta biografica di base, mappatura del contesto, approfondimento su ogni entità, ricerca di segnali negativi, incrocio tra fonti indipendenti, sintesi finale.

La parte più interessante è la scelta dei modelli. Il Direttore e i passaggi che richiedono ragionamento fine — giudizio sul rischio, mappatura delle connessioni, stesura del report — usano un modello "profondo" (Claude Opus 4 nella configurazione del progetto). L'estrazione strutturata dei fatti e il dibattito sul rischio, che sono compiti più meccanici, girano su modelli più economici (GPT-4.1, varianti Sonnet o Gemini Flash). È un compromesso esplicito tra costo e qualità: non tutto il lavoro di un'indagine richiede lo stesso livello di ragionamento, e trattarlo come se lo richiedesse significa solo spendere di più senza guadagnare precisione.

La parte operativa: cosa succede in un'indagine

Il metodo si può descrivere in cinque passaggi, applicabili concettualmente anche fuori da questo strumento specifico, in qualsiasi verifica OSINT strutturata.

1. Separare cosa cercare da come cercarlo. Il Direttore decide gli obiettivi (quali entità approfondire, quali ipotesi verificare) e li affida a nodi diversi per l'esecuzione. Nella pratica manuale, l'errore più comune è mescolare le due cose: si comincia a cercare senza aver deciso cosa si sta cercando di confermare o smentire, e la ricerca si allarga senza controllo.
2. Assegnare il costo giusto al compito giusto. Non ogni ricerca merita lo stesso sforzo. Un'estrazione

di dati strutturati (nome, ruolo, data) è un compito meccanico; giudicare se un pattern di comportamento costituisce un rischio reputazionale richiede più contesto. Trattare le due cose allo stesso modo è uno spreco, in tempo o in budget.
3. Ancorare ogni fatto a una finestra temporale, non solo a un testo. Il sistema registra non solo cosa è stato detto, ma quando è valido. Nel proprio README, il progetto documenta un esempio da una run di prova: una fonte indicava una sospensione professionale dal 2017 al 2019, un'altra la stessa persona attiva in un ruolo di consulenza dal 2017 al 2021. Le due informazioni si sovrappongono nel tempo in modo incompatibile, e il sistema le segnala come contraddizione con severità critica. È un esempio pubblicato dagli stessi sviluppatori a scopo dimostrativo, non un caso che posso verificare in modo indipendente — ma il principio è solido: senza una data associata a ogni fatto, una contraddizione del genere resta invisibile.
4. Non fidarsi di una fonte sola. Il sistema attraversa una fase dedicata solo al confronto tra fonti indipendenti prima di considerare un'informazione affidabile. È lo stesso principio della triangolazione giornalistica: una fonte, per quanto autorevole, resta un'ipotesi finché non trova conferma altrove.
5. Rappresentare le relazioni come struttura, non solo come racconto. Le connessioni tra persone, organizzazioni ed eventi vengono salvate come nodi e archi in un grafo (Neo4j, nella configurazione del progetto), non solo descritte in un testo. Questo permette di calcolare cose che un riassunto narrativo nasconde facilmente, come quali nodi sono i più connessi in una rete di relazioni.

L'errore più frequente in un'indagine open source manuale è l'opposto di tutto questo: un'unica ricerca ampia, nessuna distinzione tra dato verificato e ipotesi, nessuna traccia di quando un'informazione era valida, una fonte sola presa per buona perché sembrava autorevole.

Il risultato atteso di un processo strutturato in questo modo non è "più dati", ma dati classificati: un report che distingue i fatti confermati da quelli ancora aperti, un punteggio di rischio tracciabile fino alla fonte che lo ha generato, un grafo delle relazioni ispezionabile invece di una narrazione che va presa per fede.

Perché conta, anche fuori da questo strumento

Per un giornalista o un ricercatore OSINT che lavora da solo, replicare l'intera infrastruttura — LangGraph, più modelli, database a grafo — non ha senso per la maggior parte dei casi. Ma i cinque principi restano validi anche con un foglio di calcolo e una cronologia scritta a mano: separare la pianificazione dall'esecuzione, non spendere lo stesso sforzo su ogni domanda, datare ogni fatto, cercare conferme indipendenti prima di scrivere, e mappare le relazioni invece di limitarsi a raccontarle.

Per chi lavora in compliance o due diligence su piccola scala, il progetto è utile soprattutto come riferimento di metodo, non come strumento pronto all'uso: è un esercizio tecnico pubblicato con licenza aperta, non un prodotto testato in produzione, e il giudizio finale su un rischio reputazionale o legale resta comunque una responsabilità umana, non qualcosa che un report automatico può assorbire.

Il punto non è che un agente AI faccia la ricerca al posto tuo. È che ti costringa a rendere esplicito qualcosa che normalmente resta implicito: quando hai verificato un fatto, contro quali altre fonti, e cosa resta ancora solo un'ipotesi.
Chi fa due diligence su una persona o un'azienda conosce bene il problema centrale: non manca mai l'informazione, manca la certezza che quell'informazione sia coerente. Un ruolo professionale che su un sito risulta attivo dal 2017 e su un altro si interrompe nel 2019. Una relazione tra due enti che sembra diretta ma nasce solo da una coincidenza di date. Il lavoro di verifica open source è soprattutto questo: incrociare, contraddire, scartare.

Un progetto open source chiamato **Deep Research AI Agent** (repository ai-assessment su GitHub, sviluppato da Harsh Solanki) prova a rendere replicabile proprio questa fase, quella più lenta e più spesso saltata. Non è un prodotto commerciale: è nato come esercizio per una selezione tecnica (Deriv AI, AI Engineer Technical Assessment), pubblicato con licenza MIT. Ma il modo in cui è costruito vale la pena guardarlo da vicino, perché il metodo che applica — pianificazione separata dall'esecuzione, costo modulato sulla qualità richiesta, fatti ancorati nel tempo, relazioni

rappresentate come grafo — non riguarda solo questo strumento specifico.

Come è organizzato il lavoro

Il sistema usa **LangGraph**, un framework che permette di costruire flussi di lavoro con cicli e decisioni condizionali, non solo sequenze lineari. Al centro c'è un nodo che il progetto chiama Research Director: pianifica il passo successivo, decide quale agente specializzato attivare — ricerca web, estrazione di fatti, analisi temporale, risoluzione delle entità duplicate, analisi del rischio, mappatura delle connessioni, verifica delle fonti, generazione del report — e valuta quando fermarsi. In totale, contando anche il Direttore, sono nove nodi coordinati. La ricerca vera e propria attraversa sei fasi in sequenza: raccolta biografica di base, mappatura del contesto, approfondimento su ogni entità, ricerca di segnali negativi, incrocio tra fonti indipendenti, sintesi finale.

La parte più interessante è la scelta dei modelli. Il Direttore e i passaggi che richiedono ragionamento fine — giudizio sul rischio, mappatura delle connessioni, stesura del report — usano un modello "profondo" (Claude Opus 4 nella configurazione del progetto). L'estrazione strutturata dei fatti e il dibattito sul rischio, che sono compiti più meccanici, girano su modelli più economici (GPT-4.1, varianti Sonnet o Gemini Flash). È un compromesso esplicito tra costo e qualità: non tutto il lavoro di un'indagine richiede lo stesso livello di ragionamento, e trattarlo come se lo richiedesse significa solo spendere di più senza guadagnare precisione.

La parte operativa: cosa succede in un'indagine

Il metodo si può descrivere in cinque passaggi, applicabili concettualmente anche fuori da questo strumento specifico, in qualsiasi verifica OSINT strutturata.

1. Separare cosa cercare da come cercarlo. Il Direttore decide gli obiettivi (quali entità approfondire, quali ipotesi verificare) e li affida a nodi diversi per l'esecuzione. Nella pratica manuale, l'errore più comune è mescolare le due cose: si comincia a cercare senza aver deciso cosa si sta cercando di confermare o smentire, e la ricerca si allarga senza controllo.
2. Assegnare il costo giusto al compito giusto. Non ogni ricerca merita lo stesso sforzo. Un'estrazione di dati strutturati (nome, ruolo, data) è un compito meccanico; giudicare se un pattern di comportamento costituisce un rischio reputazionale richiede più contesto. Trattare le due cose allo stesso modo è uno spreco, in tempo o in budget.
3. Ancorare ogni fatto a una finestra temporale, non solo a un testo. Il sistema registra non solo cosa è stato detto, ma quando è valido. Nel proprio README, il progetto documenta un esempio da una run di prova: una fonte indicava una sospensione professionale dal 2017 al 2019, un'altra la stessa persona attiva in un ruolo di consulenza dal 2017 al 2021. Le due informazioni si sovrappongono nel tempo in modo incompatibile, e il sistema le segnala come contraddizione con severità critica. È un esempio pubblicato dagli stessi sviluppatori a scopo dimostrativo, non un caso che posso verificare in modo indipendente — ma il principio è solido: senza una data associata a ogni fatto, una contraddizione del genere resta invisibile.
4. Non fidarsi di una fonte sola. Il sistema attraversa una fase dedicata solo al confronto tra fonti indipendenti prima di considerare un'informazione affidabile. È lo stesso principio della triangolazione giornalistica: una fonte, per quanto autorevole, resta un'ipotesi finché non trova conferma altrove.
5. Rappresentare le relazioni come struttura, non solo come racconto. Le connessioni tra persone, organizzazioni ed eventi vengono salvate come nodi e archi in un grafo (Neo4j, nella configurazione del progetto), non solo descritte in un testo. Questo permette di calcolare cose che un riassunto narrativo nasconde facilmente, come quali nodi sono i più connessi in una rete di relazioni.

L'errore più frequente in un'indagine open source manuale è l'opposto di tutto questo: un'unica ricerca ampia, nessuna distinzione tra dato verificato e ipotesi, nessuna traccia di quando un'informazione era valida, una fonte sola presa per buona perché sembrava autorevole.

Il risultato atteso di un processo strutturato in questo modo non è "più dati", ma dati classificati: un report che distingue i fatti confermati da quelli ancora aperti, un punteggio di rischio tracciabile fino alla fonte che lo ha generato, un grafo delle relazioni ispezionabile invece di una narrazione che va

presa per fede.

Perché conta, anche fuori da questo strumento

Per un giornalista o un ricercatore OSINT che lavora da solo, replicare l'intera infrastruttura — LangGraph, più modelli, database a grafo — non ha senso per la maggior parte dei casi. Ma i cinque principi restano validi anche con un foglio di calcolo e una cronologia scritta a mano: separare la pianificazione dall'esecuzione, non spendere lo stesso sforzo su ogni domanda, datare ogni fatto, cercare conferme indipendenti prima di scrivere, e mappare le relazioni invece di limitarsi a raccontarle.

Per chi lavora in compliance o due diligence su piccola scala, il progetto è utile soprattutto come riferimento di metodo, non come strumento pronto all'uso: è un esercizio tecnico pubblicato con licenza aperta, non un prodotto testato in produzione, e il giudizio finale su un rischio reputazionale o legale resta comunque una responsabilità umana, non qualcosa che un report automatico può assorbire.

Il punto non è che un agente AI faccia la ricerca al posto tuo. È che ti costringa a rendere esplicito qualcosa che normalmente resta implicito: quando hai verificato un fatto, contro quali altre fonti, e cosa resta ancora solo un'ipotesi.