A BASSA VOCE – 12 MAGGIO 2026

Capire non basta

di Alfonso Fuggetta

L'intelligenza artificiale ci spinge verso un'analisi senza sintesi. È una perdita cognitiva che ancora non stiamo misurando.

Nell'insegnamento torno ogni anno su una distinzione che ritengo importante.

C'è una differenza fondamentale tra capire quello che fa qualcun altro e saper produrre qualcosa da soli. In aula la chiamo distinzione tra analisi e sintesi. O anche la differenza tra sapere e saper fare.

Analizzare vuol dire comprendere un programma già scritto, seguirne la logica, riconoscere gli errori. Sintetizzare significa partire da un problema e costruire la soluzione: scrivere il programma, non solo leggerlo.

Sono due capacità diverse e la sintesi non deriva automaticamente dall'analisi. Uno studente può capire perfettamente il codice che mostro alla lavagna e non riuscire a scriverne uno analogo da solo. È un'esperienza comune a chiunque insegni programmazione. Ed è, a ben pensarci, il segnale di qualcosa di più profondo e di valenza generale: sapere non equivale a saper fare.

Costruire e riconoscere chiedono cose diverse

Scrivere un programma vuol dire tenere insieme molte cose contemporaneamente: cosa devi fare, come strutturarlo, quali casi limite gestire, quali errori il compilatore segnala. Tutto va deciso esplicitamente: ogni scelta sintattica, ogni struttura di controllo, ogni gestione di un errore.

Verificare il codice generato da un LLM, o quello scritto da chiunque altro (un docente alla lavagna, un collega, un libro di testo), è un'operazione di analisi. Si scorre, si

riconosce il pattern, si dà un giudizio di plausibilità. Il codice sembra corretto, la struttura è familiare, il risultato appare ragionevole. Spesso ci si ferma lì.

Il fenomeno è preesistente, ma l'AI lo amplifica di un ordine di grandezza: la situazione in cui qualcun altro scrive il codice e io mi limito a leggerlo passa da caso particolare a modalità di lavoro principale, presente in qualsiasi momento, su qualunque problema.

C'è una tendenza naturale a risparmiare sforzo. Quando ci viene offerta una strada più breve, una risposta già confezionata, un output da approvare anziché da costruire, la prendiamo volentieri. L'AI ci offre questa strada ogni giorno.

Quello che vedo cambiare

Negli ultimi due anni vedo una tendenza a sottovalutare il ruolo della sintesi. Nelle università il problema può essere contenuto: lo studente che usa l'AI per tutto il semestre deve comunque dimostrare di saper risolvere gli esercizi all'appello d'esame (almeno nel mio caso), e questo è un presidio utile. Nel lavoro quel presidio non c'è. Si vede il codice, sembra ragionevole, si procede. I problemi emergono quando l'AI non è disponibile, quando il problema è nuovo, quando serve davvero costruire qualcosa da zero. O quando si presentano errori che costringono a entrare nel merito delle scelte effettuate.

Chi scrive il codice accumula comprensione in modo diverso da chi lo legge. Fa scelte, sbaglia, torna indietro, ricostruisce. Questo processo, faticoso, a tratti frustrante, è esattamente il meccanismo con cui le competenze si sedimentano. È il tipo di apprendimento che resiste nel tempo.

Chi invece usa l'AI per generare e si limita a verificare l'output può capire ciò che vede. Spesso capisce anche abbastanza bene. Ma quella comprensione non si trasforma in capacità produttiva, perché non ha mai dovuto costruire nulla.

Il punto non è che usare l'AI sia sbagliato. È che usarla per la produzione, riservandosi solo il ruolo di revisore, rischia di generare un profilo cognitivo sbilanciato: molta analisi, poca sintesi.

La perdita che non si vede

Il paradosso è che questa perdita è silenziosa.

Uno studente o un professionista che si affida all'AI per produrre non percepisce immediatamente la regressione. I risultati arrivano, il codice funziona, il lavoro avanza. L'effetto emerge solo dove l'AI non arriva: davanti a un problema davvero nuovo, a un vincolo che gli strumenti non hanno mai visto, alla necessità di costruire da zero senza appoggi.

È lo stesso fenomeno che si osserva con la navigazione GPS: ci si orienta benissimo finché si segue la freccia, ma si scopre di aver perso il senso dello spazio e dell'orientamento quando il segnale manca.

La capacità di costruire da zero, di tenere insieme i pezzi e di prendere decisioni esplicite passo dopo passo richiede esercizio per restare efficace. Se la si delega sistematicamente, si indebolisce, non in modo drammatico, non tutto in una volta, ma in modo costante.

E i dati empirici lo confermano. Studenti che imparano con ChatGPT, a 45 giorni di distanza, ricordano significativamente meno. Lo stesso pattern emerge nei task di programmazione: chi delega all'AI accumula meno competenza, anche quando l'obiettivo è imparare.

Una domanda che resta aperta

Non ho una risposta semplice. Né penso che la risposta sia "non usate l'AI".

Quello che penso è che dobbiamo cominciare a fare una distinzione più netta tra gli usi dell'AI che supportano la sintesi — che aiutano a costruire, suggeriscono, correggono nel corso del processo produttivo — e quelli che la sostituiscono, consegnandoci un output finito da validare. La prima modalità è potenzialmente arricchente. La seconda è cognitivamente rischiosa.

Credo che gli studenti debbano imparare a scrivere il codice da soli. Non per feticismo del passato, né per diffidenza verso gli strumenti. Ma perché capire non basta e la capacità di costruire va protetta, anche quando ogni giorno ci viene offerta una scorciatoia.

La domanda che dovremmo farci a proposito dell'AI è se stiamo formando persone con competenze di sintesi (saper fare) all'altezza, oppure solo passivi osservatori di ciò che essa produce.

La differenza non è da poco, perché incide sulla nostra capacità di dominare e sfruttare le tecnologie senza restarne vittime.

Questo post è stato scritto con l'assistenza di Claude. Le idee, le posizioni e il ragionamento sono miei.