**LISTA PAPER (FAKE NEWS DETECTION)**

PAPER 0

**An Experimental Comparison of the Most Popular Approaches to Fake News Detection**

* Pietro Dell’Oglio, Alessandro Bondielli, Francesco Marcellonia, Lucia C. Passaro
* 2025
* <https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5186254>

***METODO***

Il paper fa una **raccolta di approcci** per la Fake News Detection:

* **Machine Learning tradizionale**: veloce, interpretabile, ma limitato dalla qualità delle feature.
  + **Approcci di Machine Learning Tradizionale**: Basati su feature estratte dal contenuto o dal contesto delle notizie (es. titoli, linguaggio, utenti, reti sociali).
    - **Logistic Regression (LR)**: Modello statistico usato per compiti di classificazione binaria. Nella fake news detection sfrutta rappresentazioni testuali (spesso TF-IDF) per stimare la probabilità che una notizia sia vera o falsa
    - **Support Vector Machine (SVM)**: Cerca un iperpiano ottimale che separi notizie vere e false con il massimo margine. Anch’esso comunemente usato con TF-IDF
    - **Naive Bayes (NB)**: Modello probabilistico basato sul teorema di Bayes. È semplice ed efficace su rappresentazioni testuali sparse come TF-IDF
* **Deep Learning**: apprende rappresentazioni complesse, ma richiede molti dati e risorse.
  + **Approcci di Deep Learning (DL)**: Consentono di apprendere rappresentazioni latenti dai dati testuali, andando oltre feature manuali.
    - **CNN (Convolutional Neural Network)**: Modelli come FNDNet catturano pattern locali nelle sequenze di parole.
    - **RNN (Recurrent Neural Network), LSTM e BiLSTM**: Adatte a testi sequenziali, usate per classificare notizie o alberi di propagazione. Le varianti bidirezionali (BiLSTM) ottengono prestazioni migliori ma sono più costose.
    - **GRU (Gated Recurrent Unit)**: Alternativa alle LSTM, meno costosa ma più lenta da addestrare.
    - **Modelli Ibridi**: Combinazioni CNN+RNN o BiLSTM+CNN hanno mostrato buoni risultati.
    - **Ensemble e Explainable ML**: Alcuni lavori recenti hanno combinato più architetture o si sono concentrati sulla spiegabilità delle decisioni .
* **Transformer/LLM**: migliori prestazioni con fine-tuning, promettenti anche in zero-shot, ma ancora problemi di generalizzazione.
  + **Approcci basati su Transformer e LLM**: Questa è la linea più moderna, che sfrutta modelli linguistici pre-addestrati.
    - **BERT**: Fine-tuning per classificazione fake/real.
    - **DeBERTa**: Variante migliorata di BERT con disentangled attention e decoder potenziato; ha spesso superato BERT.
    - **FakeBERT**: Architettura CNN che usa embedding pre-addestrati di BERT, combinando DL classico e Transformer.
    - **LLM in zero-shot**: Alcuni LLM (es. LLaMA2, GPT-4o) sono usati senza fine-tuning, solo con prompt. Hanno capacità di trasferimento, ma non sempre generalizzano meglio di modelli più piccoli e specializzati.
    - **LLM come supporto**: Alcuni studi hanno mostrato che modelli come GPT-3.5 possono fornire spiegazioni multi-prospettiva utili a migliorare modelli più piccoli tramite rationale guidance.

***RISULTATI***

Dal paper emergono tre set di esperimenti (**Dataset-Specific**, **Cross-Dataset**, **Mixed-Training/Single-Test**):

* **Dataset-Specific Experiments:** Ogni modello viene addestrato e testato sullo stesso dataset.
  + **DeBERTa** è il migliore in quasi tutti i dataset. Ottiene F1 più alti e consistenti.
  + **BERT** è il secondo migliore, confermando l’efficacia del fine-tuning dei Transformer.
  + **GPT-4o** **(zero-shot)**, in alcuni dataset supera BERT (es. Celebrity, NDF, LIAR-PLUS), ma resta dietro a DeBERTa.
  + **LLaMA2 (zero-shot)** molto più debole. Senza fine-tuning fatica a competere.
  + **Tradizionali (LR, SVM, NB)** hanno risultati discreti, in alcuni dataset piccoli (NDF, Politifact) riescono anche a superare i modelli deep learning.
  + **DL classici (CNN, BiLSTM, CNN-BERT)** hanno prestazioni miste. Si comportano bene su dataset grandi (Isot, Infodemic) ma soffrono molto di overfitting e generalizzano poco.
* **Cross-Dataset Experiments:** Un modello è addestrato su un dataset e testato sugli altri (scenario di domain shift). Tutti i modelli calano drasticamente di performance per un forte problema di generalizzazione.
  + **DeBERTa** e **BERT** ancora migliori degli altri, ma con cali significativi.
  + **Modelli tradizionali (LR, SVM, NB)** in alcuni casi reggono meglio dei DL, perché meno sensibili a overfitting.
  + **DL (CNN, BiLSTM, CNN-BERT)** sono i peggiori. Fortissimo overfitting ai dati di training, generalizzano malissimo.
  + **GPT-4o e LLaMA2** restano stabili perché non si addestrano. A volte la stabilità li rende competitivi rispetto ai modelli addestrati su un singolo dataset.
* **Mixed-Training / Single-Test Experiments:** Un unico modello viene addestrato su un mix di più dataset e poi testato su set misti o su dataset tenuti fuori.
  + **DeBERTa** rimane il migliore anche qui.
  + **BERT** segue da vicino.
  + **GPT-4o** competitivo, in alcuni casi vicino a BERT.
  + **Tradizionali (LR, SVM, NB)** e **DL classici** restano più indietro, anche se migliorano grazie alla varietà del training.

Quindi il **fine-tuning di Transformer** (soprattutto DeBERTa) è il miglior compromesso tra performance e generalizzazione.

**LLM zero-shot (**GPT-4o, LLaMA2) hanno potenziale per contesti dove mancano dati annotati. Sono interessanti per scenari reali, ma ancora non stabili.

**Modelli tradizionali** sorprendono per solidità su dataset piccoli o specifici, e mantengono una certa competitività.

**Deep Learning classico (CNN, BiLSTM)** soffre molto di overfitting e mostra limiti di generalizzazione.

PAPER 9

**Automatic Detection of Fake News**

* Veronica Perez-Rosas, Bennett Kleinberg, Alexandra Lefevre, Rada Mihalcea
* 2017
* <https://arxiv.org/abs/1708.07104>

NOTE: Paper dove viene usato il dataset “Celebrity”

***METODO***

Gli autori hanno usato **principalmente caratteristiche linguistiche** (no deep learning, solo ML classico), considerando N-grams, punteggiatura, LIWC features, readability metrics e sintassi. Come classificatore usato un **SVM lineare** (Support Vector Machine).

* **Estrazione delle feature**
  + Le notizie (vere e false) sono state trasformate in rappresentazioni numeriche basate su diverse famiglie di feature linguistiche:
    - **N-grams** (unigrammi e bigrammi): rappresentazione Bag-of-Words con pesi tf-idf.
    - **Punteggiatura** → conteggi di 11 simboli (es. ., ,, ?, !).
    - **LIWC features**: percentuali di parole in categorie psicologiche/linguistiche (emozioni, pronomi, processi cognitivi, ecc.).
    - **Readability metrics**: indici di leggibilità e complessità del testo (Flesch-Kincaid, Gunning Fog, lunghezza media frasi, parole complesse).
    - **Sintassi**: regole di produzione estratte dagli alberi sintattici (via Stanford Parser), anch’esse pesate con tf-idf.
    - **Combinazione di tutte le feature**: vettore unico con oltre 2000 feature.
* **Classificatore**: **Support Vector Machine (SVM)**
  + Per ogni esperimento, si addestrava il classificatore su un sottoinsieme di dati e si testava su un altro.
  + Validazione incrociata a 5 fold (**5-fold cross-validation**). Garantisce che ogni notizia sia usata sia per training che per test in momenti diversi.
* **Funzionamento pratico**
  + Ogni articolo viene trasformato in un vettore di feature linguistiche.
  + L’SVM cerca un iperpiano lineare che separi meglio gli articoli veri dai falsi nello spazio delle feature.
  + Il modello impara quali combinazioni di **caratteristiche sono più indicative di fake news** (es. uso eccessivo di verbi, pronomi sociali, punteggiatura) rispetto a quelle legittime (es. processi cognitivi, linguaggio analitico).

***RISULTATI***

* **FakeNewsAMT** (crowdsourced, multi-dominio):
  + Migliori feature: Readability → 78% accuracy.
  + Tutte le feature combinate → 74% accuracy.
* **Celebrity dataset** (dal web):
  + Migliori feature: Punteggiatura → 70% accuracy.
  + Tutte le feature combinate → 73% accuracy.
* **Cross-domain**:
  + Addestrare su un dataset e testare sull’altro, porta a crolli drastici (accuracy ~0.50–0.61). Segno che le caratteristiche della scrittura fake sono molto dipendenti dal dominio.

Fake news tendono a usare più parole sociali e positive, più certezza, più verbi e avverbi, e più punteggiatura. Invece, notizie vere tendono a includere più processi cognitivi (insight, differenziazione, tentativi), pronomi funzionali (he, she, negazioni), linguaggio analitico.

***DATASET***

Gli autori hanno creato due corpora:

* **FakeNewsAMT** (via crowdsourcing su Amazon Mechanical Turk)
  + 240 articoli reali raccolti da fonti mainstream USA (CNN, NYTimes, Bloomberg, ecc.) in 6 domini: sport, business, intrattenimento, politica, tecnologia, educazione.
  + 240 articoli fake scritti dai lavoratori AMT a partire da brevi estratti degli articoli veri.
  + Totale: ~33k parole vere, ~32k parole false.
* **Celebrity** dataset (raccolto dal web)
  + 100 articoli reali + 100 fake relativi a celebrità (attori, cantanti, politici).
  + Fake presi da tabloid o siti di gossip (es. RadarOnline), verificati contro siti di fact-checking (es. GossipCop).
  + Le notizie false riguardavano soprattutto gossip sensazionalistici (divorzi, tradimenti, ecc.).
  + Totale: ~40k parole fake, ~71k parole vere.

PAPER 10

**Disinformation Detection about Islamic Issues on Social Media using Deep Learning Techniques**

* Suhaib Kh. Hamed1, Mohd Juzaiddin Ab Aziz, Mohd Ridzwan Yaakub
* 2023
* <http://mjs.um.edu.my/index.php/MJCS/article/view/41935>

NOTE: Paper del dataset “CIDII”

***METODO***

Il modello proposto è basato su **Bi-LSTM** (Bidirectional Long Short-Term Memory), con l’uso di **word** **embeddings personalizzati**. Funziona in tre fasi principali:

* **Embedding Layer**: Embedding Layer con GloVe pre-addestrato (300 dimensioni), poi ri-addestrato (Mittens) su un corpus islamico (per gestire parole fuori vocabolario, OOV).
  + Ogni parola del testo viene trasformata in un **vettore numerico** tramite **GloVe** (Global Vectors for Word Representation).
  + Molti termini islamici (es. Hadith, Sharia, Jizya) non erano nel vocabolario di GloVe e quindi venivano trattati come OOV (Out Of Vocabulary). Come soluzione, gli autori hanno ri-**addestrato GloVe** su un **corpus islamico** (quasi 1M di parole) usando il metodo Mittens, che aggiorna i vettori pre-esistenti adattandoli al dominio specifico.
  + In questo modo, gli embedding sono più adatti al linguaggio religioso/islamico, mantenendo coerenza con i vettori originali.
* **Bi-LSTM Layer**:
  + Il cuore del modello è una **rete ricorrente bidirezionale**:
    - Una LSTM legge la frase da sinistra a destra.
    - Una seconda LSTM legge la frase da destra a sinistra.
  + Questo consente di catturare sia il contesto passato sia quello futuro delle parole.
* **Dense Layer + Sigmoid**: Dense Layer con Sigmoid per classificazione binaria (corretto vs disinformazione).
  + Dopo la Bi-LSTM, i vettori passano a un fully connected layer.
  + L’uscita finale usa una funzione di attivazione Sigmoid, e restituisce un valore tra 0 e 1 (informazione corretta / disinformazione)

***RISULTATI***

Confronto tra modelli (con GloVe ri-addestrato su corpus islamico):

* Bi-LSTM: AUC 95.42% | F1-score 97.31%.
* GRU: AUC 90.81%.
* CNN: AUC 92.78%.
* LSTM: AUC 84.62%.

Quindi, **Bi-LSTM + GloVe ri-addestrato** (Mittens) è la migliore combinazione per questo dominio.

***DATASET***

**CIDII** (Correct Information and Disinformation about Islamic Issues).

* Dimensioni: 731 esempi totali.
* 431 = informazioni corrette.
* 300 = disinformazione.

PAPER 11

**This Just In: Fake News Packs a Lot in Title, Uses Simpler, Repetitive Content in Text Body, More Similar to Satire than Real News**

* Benjamin D. Horne, Sibel Adali
* 2017
* <https://ojs.aaai.org/index.php/ICWSM/article/view/14976>

NOTE: Paper del dataset “Horne”

***METODO***

Il modello utilizzato è una **Support Vector Machine (SVM)** con kernel lineare. In input al modello **non vengono date tutte le feature**, ma **solo le 4 più significative** (separate per titolo e per corpo).

Gli autori hanno estratto tre categorie di feature da titoli e corpi degli articoli

* **Stylistic features** (struttura del testo):
  + POS tagging (nouns, verbs, adverbs, proper nouns)
  + Stopwords, punteggiatura, uso di MAIUSCOLE, pronomi personali, ecc.
* **Complexity features** (complessità linguistica):
  + Lunghezza media frasi/parole
  + Profondità degli alberi sintattici (Stanford Parser)
  + Readability indexes (Gunning Fog, SMOG, Flesch-Kincaid)
  + Lexical diversity (Type-Token Ratio, TTR)
  + Frequenza dei termini nel Corpus of Contemporary American English (fluency)
* **Psychological features** (processi cognitivi ed emotivi):
  + LIWC (categorie come analytic words, insight, certainty, power, emotion, ecc.)
  + SentiStrength (intensità di emozioni positive/negative)

***RISULTATI***

**Fake news più brevi**, meno complesse, e più ridondanti delle real. Usano meno punteggiatura, meno citazioni, più pronomi personali, meno parole analitiche e meno termini tecnici. Risultano più simili alla satira che al giornalismo reale.

Le differenze più marcate stanno nei **titoli**. Fake hanno titoli più lunghi, con meno stopwords, meno nomi comuni, ma più nomi propri, verbi e maiuscole. Invece, le real news tendono ad avere titoli concisi.

**Fake e satire condividono molte caratteristiche** (semplicità, ridondanza, linguaggio meno tecnico). Fake differisce da satire solo perché esagera le stesse deviazioni dal giornalismo reale.

Performance del modello (SVM)

* **Corpo articoli**:
  + Fake vs Real: 71–77% accuracy (baseline ~50%)
  + Satire vs Real: 91%
  + Satire vs Fake: solo 67% (più difficile separarli).
* **Titoli**:
  + Fake vs Real: 71–78%
  + Satire vs Real: 75%
  + Satire vs Fake: 55% (quasi random).

***DATASET***

* Buzzfeed 2016 US Election dataset
  + 36 articoli reali, 35 articoli fake.
  + Selezionati da Buzzfeed in base all’alta condivisione su Facebook.
* Dataset creato dagli autori (political news)
  + 75 articoli reali, 75 fake, 75 satirici.
  + Raccolti da fonti note (Wall Street Journal, Infowars, The Onion, ecc.).
  + Tutti “hard news”, niente opinioni.
* Dataset Burfoot & Baldwin (2009)
  + 4000 real news, 233 satire news.
  + Usato per rafforzare le comparazioni.

PAPER 12

**Fighting an Infodemic: COVID-19 Fake News Dataset**

* Parth Patwa, Shivam Sharma, Srinivas PYKL, Vineeth Guptha, Gitanjali Kumari, Md Shad Akhtar, Asif Ekbal, Amitava Das, Tanmoy Chakraborty
* 2021
* <https://link.springer.com/chapter/10.1007/978-3-030-73696-5_3>

NOTE: Paper del dataset “Infodemic”

***METODO***

L’obiettivo principale del paper è quello di testare il dataset su una serie di baseline models per valutarlo, seguendo questi punti:

* **Pre-processing del testo**: Rimozione di link, simboli non alfanumerici, hashtag non informativi. Eliminazione delle stopwords (parole comuni come “the”, “and”, “is”).
* **Feature Extraction – TF-IDF**: Ogni testo è rappresentato tramite un **vettore numerico** che misura l’importanza delle parole. In questo modo, una parola ha più peso se compare spesso in un documento, ma meno se è comune in tutto il corpus.
* **Classificatori utilizzati:** Gli autori hanno testato 4 modelli classici di machine learning (usando scikit-learn):
  + **Decision Tree (DT)**: modelli ad albero, semplici ma tendono a sovradattare.
  + **Logistic Regression (LR)**: modello lineare che separa fake/real in base a una funzione logistica.
  + **Gradient Boosted Decision Trees (GDBT)**: combinazione di più alberi per migliorare le prestazioni.
  + **Support Vector Machine (SVM, kernel lineare)**: cerca un iperpiano di separazione ottimale che massimizzi il margine tra notizie fake e real.

Dallo studio, **l’SVM funziona meglio**. Le fake news e le real news, pur condividendo vocabolario, presentano pattern linguistici diversi. L’SVM con kernel lineare è particolarmente adatto a dati ad alta dimensionalità e sparsi come quelli TF-IDF (dove ogni parola è una dimensione).

***RISULTATI***

Migliori performance:

* SVM: F1 = 93.32% (test set)
* Logistic Regression: F1 = 91.96%

Prestazioni inferiori:

* Decision Tree (85.39% F1)
* Gradient Boost (86.96% F1)

***DATASET***

**Infodemic**: totale di 10.700 testi (tweet, post, articoli).

* **Real news**: raccolte da account Twitter verificati (es. OMS, CDC, ICMR, testate giornalistiche ufficiali).
* **Fake news**: raccolte da siti di fact-checking (PolitiFact, Boomlive, Snopes, NewsChecker, IFCN chatbot, Google Fact Check Explorer), oltre a post da Facebook, Twitter, Instagram.

PAPER 13

**Detection of Online Fake News Using N-Gram Analysis and Machine Learning Techniques**

* Hadeer Ahmed, Issa Traore, Sherif Saad
* 2017
* <https://www.researchgate.net/publication/320300831_Detection_of_Online_Fake_News_Using_N-Gram_Analysis_and_Machine_Learning_Techniques>

NOTE: Paper del dataset “ISOT”

***METODO***

Il modello è basato su un pipeline classico di text classification, strutturato in 4 fasi principali:

* **Pre-processing del testo:** Gli articoli vengono puliti e trasformati in una forma standard ( ridurre il rumore e la dimensionalità)
* **Rappresentazione del testo (Feature Extraction con n-grammi):**
  + Il testo non viene usato così com’è, ma trasformato in vettori numerici tramite n-grammi di parole:
    - Unigrammi (singole parole)
    - Bigrammi (sequenze di due parole consecutive)
    - Trigrammi e Four-grammi
  + Due metodi di pesatura testati:
    - **TF (Term Frequency)**: conta normalizzata delle parole.
    - **TF-IDF (Term Frequency-Inverse Document Frequency)**: pesa di più le parole distintive e riduce il peso delle parole troppo frequenti.
* **Classificazione (Machine Learning):** Sui vettori così ottenuti vengono addestrati diversi classificatori supervisionati per distinguere articoli “fake” vs “true”.
  + Linear SVM (LSVM)
  + SVM classico
  + Logistic Regression
  + SGD (Stochastic Gradient Descent)
  + Decision Tree
  + K-Nearest Neighbors

***RISULTATI***

Dalle analisi è risultato che:

* **Linear-based classifiers** (LSVM, SGD, LR) hanno dato i risultati migliori.
* **LSVM + TF-IDF (unigram)** → miglior accuratezza: 92%.
* **DT** ha raggiunto fino all’89%.
* **SVM classico** e **KNN** hanno performato peggio, specialmente con n-gram > 2.
* **TF-IDF** ha superato nettamente **TF**.
* L’aumento della dimensione n-gram oltre il unigram ha peggiorato l’accuratezza.

Il modello più efficace per la detection di fake news è **Unigram + TF-IDF + Linear SVM**, con 92% accuratezza. Inoltre, si è notato che i classificatori lineari sono i più robusti, e che gli n-gram più lunghi (bigram, trigram) non migliorano la performance.

***DATASET***

**ISOT**: dataset bilanciato, principalmente articoli politici legati alle elezioni USA del 2016.

* 12.600 articoli veri presi da Reuters.
* 12.600 articoli falsi presi da un dataset Kaggle (fake news provenienti da siti inaffidabili segnalati da Politifact).

PAPER 14

**Where is your Evidence: Improving Fact-checking by Justification Modeling**

* Tariq Alhindi, Savvas Petridis, Smaranda Muresan
* 2018
* <https://aclanthology.org/W18-5513/>

NOTE: Paper del dataset “LIAR-PLUS”

***METODO***

Gli autori vogliono migliorare i modelli automatici di fact-checking, che fino ad allora si basavano quasi esclusivamente sulla frase della dichiarazione (claim) e su metadati del parlante, senza sfruttare l’evidenza (justification) che i fact-checker umani usano quando assegnano le etichette di veridicità. Gli autori testano sia **modelli tradizionali** che **deep learning**, sotto 4 condizioni di input:

* **S (statement)**: unigrammi (parole singole) della frase del claim.
* **S+M (statement + metadata)**: oltre ai unigrammi, aggiungono:
  + Sentiment (SentiStrength) → punteggio di positività/negatività.
  + Emozioni (NRC EmoLex) → otto emozioni di base.
  + LIWC → tratti linguistici (es. uso di pronomi, parole cognitive).
  + Metadata del parlante (es. partito, storico delle dichiarazioni passate).
* **SJ** e **S+MJ**: aggiungono la giustificazione (con unigrammi estratti dal testo PolitiFact).

Come **modelli tradizionali** sono stati testati:

* Logistic Regression (LR)
* Support Vector Machines (SVM) con kernel lineare
  + Input: unigrammi, tf-idf, o GloVe embeddings (migliore: unigrammi).
  + Con metadati: aggiunte features di sentiment (SentiStrength), emozioni (NRC EmoLex), linguistica (LIWC), storico speaker.

Per il **Deep Learning** sono stati testate:

* **BiLSTM** **con embeddings GloVe** (100-dim).
  + Ogni parola viene rappresentata con embeddings GloVe (100 dimensioni).
  + Il claim (e, se incluso, la giustificazione) viene passato a un BiLSTM che legge la sequenza sia da sinistra che da destra.
  + L’output finale del BiLSTM va in un softmax layer per predire la classe (true, false, ecc.).
* **Parallel BiLSTM (P-BiLSTM)**: un BiLSTM per il claim e un altro per la giustificazione.
  + Invece di concatenare claim e giustificazione in un unico testo, usa due BiLSTM separati: uno legge il claim, l’altro legge la giustificazione.
  + Le due rappresentazioni vengono concatenate e passate al softmax.

***RISULTATI***

Per le analisi sono state testati questi due task

* **Binary classification**: TRUE vs FALSE (accorpando le 6 etichette in 2 gruppi).
* **Six-way classification**: tutte le 6 etichette.

I risultati principali sono

* **Senza justification** (S o S+M): F1 medio ≈ 0.23–0.27 (six-way).
* **Con justification** (SJ o S+MJ): F1 medio ≈ 0.34–0.38 (six-way).

Se usi solo il claim (S condition), la performance è bassa (F1 ≈ 0.23–0.26 nella classificazione a 6 etichette). Tra i deep learning, il BiLSTM e il P-BiLSTM sono i migliori, ma la differenza fra loro è piccola.

Quindi non serve un modello super sofisticato, basta aggiungere le giustificazioni e le performance migliorano nettamente.

***DATASET***

**LIAR dataset**:

* 12.836 dichiarazioni politiche con etichetta di veridicità in sei classi: pants-fire, false, mostly-false, half-true, mostly-true, true.

**LIAR-PLUS** (estensione proposta dagli autori):

* Per ogni claim, gli autori hanno estratto automaticamente la giustificazione dalle sezioni finali degli articoli di PolitiFact (“our ruling” o “summing up”).
* Se non c’era, hanno preso le ultime 5 frasi dell’articolo (escludendo la frase con il verdetto).
* La giustificazione contiene frasi che supportano o contraddicono il claim.

PAPER 15

**Fake News Detection on Social Media: A Data Mining Perspective**

* Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, Huan Liu
* 2017
* <https://dl.acm.org/doi/abs/10.1145/3137597.3137600>

NOTE: Paper del dataset “Politifact”

***METODO***

Il lavoro presenta diversi approcci, suddivisi in due grandi categorie:

* **Content-based models** (si basano sul contenuto della notizia)
  + **Knowledge-based**: fact-checking automatico, confrontando affermazioni con fonti esterne
  + **Style-based**: analisi dello stile linguistico per cogliere segnali di manipolazione:
    - **Deception-oriented**: identificare tracce di linguaggio ingannevole (es. sintassi atipica, regole grammaticali, reti neurali come CNN per testi).
    - **Objectivity-oriented**: rilevare scrittura sensazionalistica, clickbait, iper-partigiana.
* **Social context models** (sfruttano l’interazione degli utenti)
  + **Stance-based**: valutano le opinioni/stance degli utenti (supporto, rifiuto, neutralità) nei post e nei commenti per stimare la veridicità della notizia.
  + **Propagation-based**: analizzano le reti di diffusione (chi segue chi, come si diffonde la notizia) per inferire la credibilità. Si usano modelli di grafo e propagazione simili a PageRank o network optimization.

Quindi quello che viene analizzato sono:

* **Feature di contenuto** (linguistiche, stilistiche, visive).
* **Feature di contesto sociale** (utenti, post, reti di diffusione).
* Classificatori supervisionati standard (**SVM, Naïve Bayes, Decision Tree, KNN**) o più complessi (**ensemble, probabilistici, deep learning con CNN/RNN**).

***RISULTATI***

I modelli **solo basati sul contenuto** sono **insufficienti**: lo stile linguistico del fake news imita quello reale. Le **feature sociali** (propagazione, profilo degli utenti, stance) **migliorano** significativamente le performance.

Quindi c’è la necessità di **dataset più completi** (contenuto + contesto sociale), l’interesse verso **early** **detection** (intercettare fake news nelle prime fasi di diffusione), e l’importanza di **modelli semi-supervised o unsupervised** (dato che le etichette sono costose da ottenere).

***DATASET***

* **BuzzFeedNews (2016)**: articoli Facebook verificati da giornalisti.
* **LIAR (PolitiFact)**: 12.836 frasi etichettate in sei categorie (“pants-fire” → “true”).
* **BS Detector**: etichette automatiche basate su domini sospetti.
* **CREDBANK**: 60 milioni di tweet valutati da annotatori crowd (non specificamente fake news, ma credibilità di eventi).

PAPER 16

**In-context annotation of topic-oriented datasets of fake news: A case study on the notre-dame fire event**

* Lucia C. Passaro, Alessandro Bondielli, Pietro Dell’Oglio, Alessandro Lenc, Francesco Marcelloni
* 2022
* <https://www.sciencedirect.com/science/article/abs/pii/S0020025522008167>

NOTE: Paper del dataset “NDF”

***METODO***

L’idea centrale è introdurre **l’in-context annotation** (IC): invece di chiedere agli annotatori di decidere se una notizia sia vera o falsa senza alcun aiuto (out-of-context, OOC), si fornisce loro un set di fake seed news già conosciute (cioè bufale verificate da servizi come Snopes). In questo modo l’annotazione si avvicina di più a quella fatta da esperti fact-checker.

* Gli esperimenti si basano sul modello **Transformer BERT** (bert-base-uncased): Viene usato in modalità fine-tuning, cioè riaddestrato su un piccolo dataset specifico (Notre-Dame Fire o PHEME), con l’obiettivo di classificare una notizia come Fake o Real.
* Sono stati addestrati **due classificatori BERT**, uno con **dataset OOC** e uno con **dataset IC**, entrambi valutati contro i dati MFC (gold).
* Il modello è strutturato così:
  + **Input**: il testo della notizia (tweet o articolo).
  + **BERT encoder**: trasforma il testo in una rappresentazione vettoriale (embedding contestuale).
  + **Classification head**: un livello fully connected sopra l’output [CLS] di BERT, che mappa la rappresentazione in due classi: Real o Fake.

***RISULTATI***

Il modello è stato **valutato rispetto alle etichette MFC (gold standard)**, per capire quale metodo di annotazione (OOC vs IC) produce dati più utili per l’addestramento.

* Risultati principali (Notre-Dame Fire dataset):
  + Accuratezza simile (≈0.82) per entrambi i modelli.
  + Differenze sulla classe Fake (più critica):
    - Recall Fake:
      * OOC = 0.693
      * IC = 0.799
    - F1 Fake:
      * OOC = 0.682
      * IC = 0.725
  + Significa che il modello addestrato con **annotazioni IC riconosce meglio** le notizie false, riducendo i falsi negativi (cioè fake news scambiate per vere).

L’IC annotation migliora la qualità dei dataset crowdsourced, avvicinandosi molto al lavoro degli esperti. Permette di ottenere modelli più efficaci nel riconoscere fake news, soprattutto quando i dati sono pochi e quando la classe Fake è minoritaria.

Rimangono difficili da rilevare alcune fake news molto sottili/sofisticate (sia per gli umani che per i modelli).

***DATASET***

* **Notre-Dame Fire Dataset (NDF)**: Raccoglie 568 testi (481 tweet + 87 articoli) prodotti durante l’incendio della cattedrale di Notre-Dame (aprile 2019).
  + Ogni testo è stato annotato in tre modi:
    - OOC – crowdsourcing senza contesto.
    - IC – crowdsourcing con lista di fake news note sul tema.
    - MFC (Manually Fact-Checked) – fact-checking manuale dagli autori (gold standard).
    - Dataset disponibile su GitHub: Unipisa/NDFDataset.
* **PHEME dataset**
  + Originariamente raccolto per la rilevazione di rumour.
  + Gli autori hanno simulato un contesto fake news prendendo l’evento Charlie Hebdo e costruendo un set bilanciato di 262 tweet (131 reali, 131 fake), annotati sia OOC che IC per confrontare le performance.

PAPER 17

**GraphRAG-Causal: A Novel Graph-Augmented Framework for Causal Reasoning and Annotation in News**

* Abdul Haque, Umm e Hani, Ahmad Din, Muhammad Babar, Ali Abbas, Insaf Ullah
* 2025
* <https://arxiv.org/abs/2506.11600v1>

NOTE: Non considerano direttamente il testo ma fanno altro

***METODO***

Il framework è strutturato in tre fasi principali:

* **Data Preparation:** Raccolta di frasi dal dataset, e annotazione manuale di frasi con cause, effect e trigger. Si fa la **conversione delle frasi annotate in causal graphs** (nodi = eventi, cause, effetti, trigger; archi = relazioni), e poi si fa l’inserimento dei grafi in un database Neo4j.
  + Si parte dai dati grezzi (frasi del Causal News Corpus). Gli autori hanno preso circa 2000 frasi e ne hanno annotate manualmente 1030, etichettando:
    - **Cause**: la ragione di un evento (es. “piogge torrenziali”).
    - **Effect**: la conseguenza (es. “inondazioni”).
    - **Trigger**: parole che segnalano un legame causale (es. “a causa di”, “perché”, “provocando”).
  + Queste annotazioni sono state trasformate in grafi causali. I **nodi** sono cause, effetti, trigger, mentre gli **archi** sono relazioni (“CAUSES”, “RESULTS\_IN”, “HAS\_TRIGGER”).
  + Infine, i grafi sono stati salvati in **Neo4j**, un database a grafo, così da poter fare query complesse.
* **Graph Retrieval:** Ogni evento viene rappresentato anche come embedding. Viene usata una query ibrida che combina similarità semantica e informazioni strutturali (presenza di cause, effetti, trigger nel grafo), e dopo aver calcolato uno hybrid score allora si recupera uno dei top-k eventi più rilevanti dal database per usarli come esempi few-shot.
  + Data una nuova frase di notizia, bisogna trovare esempi simili già nel database. Siccome ogni evento è rappresentato anche come **embedding** (vettore numerico che cattura il significato semantico) con il modello **all-MiniLM-L6-v2** di HuggingFace (384 dimensioni), quando arriva una nuova frase:
    - Viene trasformata in embedding.
    - Si fa una ricerca ibrida:
      * **Semantic similarity**: confronto dell’embedding della frase con quelli già nel database (cosine similarity).
      * **Structural score**: controlla se nel grafo ci sono cause/effetti/trigger collegati (cioè se la frase ha un “vero” legame causale strutturato).
    - Si calcola uno score:
      * Gli eventi con punteggio più alto vengono recuperati e passati allo step successivo.
  + Il sistema recupera esempi più pertinenti, non solo simili nelle parole, ma anche nel tipo di relazioni causali.
* **LLM Inference**: I top-k esempi vengono inclusi in un prompt XML few-shot. Poi viene usato l’LLM DeepSeek R1 Distill LLaMA-70B (in altri test, LLaMA 4 Maverick 17B Instruct 128B). Il modello deve restituire output strutturato in JSON con frase etichettata e label causale (1 = causale, 0 = non causale). Se causalità è presente, il sistema visualizza anche il grafo causale.
  + I top-k esempi recuperati vengono messi dentro al prompt, in formato **XML few-shot**. L’uso dell’XML serve a dare **regole rigide** (formato standard) e a guidare l’LLM a produrre sempre output coerente.
  + Gli LLMs usati nei test:
    - **DeepSeek R1 Distill LLaMA-70B** (tramite Groq API, molto potente ma distillato quindi più leggero).
    - **LLaMA 4 Maverick 17B Instruct 128B** (per confrontare le performance).
  + L’LLM riceve la frase nuova con gli esempi e deve restituire un JSON con due campi:
    - tagged\_sentence (frase annotata con cause/effetti/trigger),
    - label (1 se la frase è causale, 0 altrimenti).
  + Se il risultato è causale, viene mostrata anche la visualizzazione del grafo corrispondente.
  + In questa fase, l’LLM non lavora “alla cieca”, ma supportato da esempi rilevanti e da un formato strutturato

***RISULTATI***

Con 20 esempi few-shot, GraphRAG-Causal ottiene:

* F1-score = 0.8216
* Accuracy ~ 79.6%
* Precision ~ 0.79
* Recall ~ 0.85

Con più esempi (fino a 50) le performance rimangono stabili intorno a F1 ~ 0.81–0.83.

Risultati comparabili (e in alcuni casi migliori) rispetto a modelli BERT/RoBERTa fine-tuned sullo stesso dataset.

***DATASET***

**Causal News Corpus (CNC)**

* Totale frasi raccolte: 2005
* Annotate manualmente: 1030

PAPER 18

**Towards Self-Supervised Cross-Domain Fake News Detection**

* Carmela Comito, Francesco Sergio Pisani, Erica Coppolillo, Angelica Liguori, Massimo Guarascio, Giuseppe Manco
* 2023
* <https://ceur-ws.org/Vol-3488/paper12.pdf>

***METODO***

L’obiettivo del paper è creare un **modello cross-domain**, capace di riconoscere notizie false anche in **domini nuovi**, senza etichette specifiche.

L’architettura è end-to-end e combina tre componenti principali:

* **Feature Extractor BERT** pre-addestrato (Wikipedia) per trasformare i testi in embeddings. In futuro, può integrare dati multi-modali (immagini, grafi sociali, ecc.).
  + Il testo passa in BERT, un modello di linguaggio pre-addestrato, che lo trasforma in una rappresentazione numerica (embedding).
* **Fake News Detector,** consiste in due sottoreti:
  + **Generalization Network (G)**: rimuove le caratteristiche specifiche del dominio, **generando embeddings “domain-invariant”**.
    - Riceve l’embedding da BERT.
    - Ha il compito di “ripulirlo” togliendo i dettagli che sono tipici di un dominio specifico (es. parole tipiche della politica vs gossip).
    - Il risultato è un embedding “più universale”, cioè valido anche su domini diversi.
  + **Classifier (C)**: un classificatore binario che **distingue fake/real sulle features generalizzate**.
    - Usa l’output di G (le features “neutre”) per decidere se la notizia è vera o falsa.
    - È un normale classificatore binario (output 0/1).
* **Domain Discriminator (A)** che serve come **“avversario” per forzare la rete a eliminare tracce del dominio**. Il suo compito è capire da quale dominio arriva la notizia (es. politica o gossip). Funziona come una rete di controllo: se riesce a distinguere bene il dominio, significa che G non sta facendo bene il suo lavoro. Sono state testate due varianti:
  + **CL (Classifier-based)**: usa un **classificatore sul dominio**, e quindi richiede che i dati siano etichettati col dominio di origine.
    - A è un normale classificatore: riceve le features e cerca di etichettare il dominio.
    - Richiede che i dati abbiano già un’etichetta di dominio (politica/gossip).
    - Meno flessibile, perché non sempre queste etichette sono disponibili.
  + **AE (Autoencoder-based)**: usa un **autoencoder**, e quindi non richiede etichette di dominio, più flessibile e self-supervised.
    - A è un autoencoder: invece di avere etichette, cerca di ricostruire le features.
    - L’idea è che durante la ricostruzione impari una rappresentazione del dominio, senza bisogno di etichette.
    - Più self-supervised e quindi più adatto a dati nuovi o non etichettati.
* il modello funziona come una **rete avversaria**: G cerca di ingannare A (che tenta di riconoscere il dominio), obbligando così G a produrre features neutrali rispetto al dominio, ma ancora utili per il detector C.

***RISULTATI***

Viene considerata come baseline un MLP semplice con embeddings da BERT. Dai risultati ottenuti (cross-domain) si nota che

* Entrambe le varianti (CL e AE) superano la baseline.
* L’AE (autoencoder) ottiene i risultati migliori in generalizzazione cross-domain, soprattutto GossipCop → Politifact.
* L’uso di domain adversarial learning migliora la robustezza contro domini non visti.

***DATASET***

* **PolitiFact** (politica): 814 articoli (~818 parole in media).
* **GossipCop** (gossip/celebrità): 4719 articoli (~349 parole in media).

PAPER 19

**“Liar, Liar Pants on Fire”: A New Benchmark Dataset for Fake News Detection**

* William Yang Wang
* 2017
* <https://aclanthology.org/P17-2067/>

NOTE: sembra che usi altre informazioni oltre al testo

***METODO***

Il paper valuta sia metodi classici sia modelli di deep learning:

* **Baseline**: majority class
* **Metodi tradizionali**:
  + Logistic Regression (LR)
  + Support Vector Machines (SVM)
* **Deep learning**:
  + Bi-directional LSTM (Bi-LSTM)
  + CNN (Kim, 2014)
  + **Hybrid CNN** proposto nel paper: combina testo + metadati del parlante.

Architettura dell’**Hybrid CNN**:

* rappresentazione testuale ottenuta da un CNN con max-pooling,
* rappresentazione dei metadati tramite embedding + convoluzione + Bi-LSTM,
* concatenazione delle due rappresentazioni → fully connected + softmax per la classificazione finale.

Il modello proposto **Hybrid Convolutional Neural Network (CNN)** combina due fonti di informazione:

* **testo dell’affermazione**: Viene rappresentato come sequenza di embedding delle parole (inizializzati con word2vec pre-addestrati su Google News, 300 dimensioni).
  + Questi embedding passano attraverso una CNN per il testo, ottenendo così una rappresentazione densa dell’intera frase.
* **metadati del parlante e del contesto**: Ogni informazione (es. partito, stato, lavoro, cronologia di affermazioni, ecc.) viene codificata in embedding.
  + Questi embedding passano attraverso una CNN separata e poi un Bi-LSTM che cattura le dipendenze tra i diversi metadati.
  + In questo modo, si ottiene una rappresentazione vettoriale che tiene conto di correlazioni tra, ad esempio, l’affiliazione politica e la cronologia di dichiarazioni false/vere di un parlante.
* **Fusione delle rappresentazioni**: la rappresentazione del testo (dal CNN per le frasi) e quella dei metadati (dal CNN + Bi-LSTM) vengono concatenate. Il vettore risultante viene passato a uno strato fully connected con softmax, che produce la **classificazione finale su 6 classi di veridicità** (pants-fire, false, barely-true, half-true, mostly-true, true).

***RISULTATI***

Risultati principali sul test set:

* Majority: 0.208
* SVM: 0.255
* Logistic Regression: 0.247
* Bi-LSTM: 0.233 (scarso, per overfitting)
* CNN: 0.270 (migliore tra i text-only)
* Hybrid CNN (text + all metadata): 0.274 (migliore complessivo)

**Conclusione**: il CNN semplice è già molto forte; l’inclusione dei metadati migliora ulteriormente, anche se in maniera contenuta.

***DATASET***

**LIAR**: 12.836 affermazioni brevi raccolte da PolitiFact.com (coprono circa un decennio, 2007–2016).

* I giudizi sono basati su analisi giornalistiche di esperti e corredati da fonti e spiegazioni (non generati tramite crowdsourcing).

PAPER 20

**FakeNewsNet: A Data Repository with News Content, Social Context and Spatialtemporal Information for Studying Fake News on Social Media**

* Kai Shu, Deepak Mahudeswaran, Suhang Wang, Dongwon Lee, Huan Liu
* 2018
* <https://arxiv.org/abs/1809.01286>

NOTE: due modelli usano anche le interazioni tra utenti

***METODO***

Gli autori hanno confrontando modelli di machine learning tradizionali e modelli basati su deep learning, con tre approcci:

* **Solo contenuto**: modelli basati solo sul contenuto della notizia. Rappresentano il testo con una bag-of-words (one-hot encoding).
  + SVM, Logistic Regression, Naive Bayes, CNN.
  + **Social Article Fusion (SAF/S)**: un **autoencoder** che impara rappresentazioni dal testo e poi classifica le notizie come vere o false.
* **Solo contesto sociale**:
  + **SAF/A**: modella i pattern temporali di engagement con **LSTM**. Variante che considera il pattern temporale delle interazioni degli utenti (tweet, retweet, reply, like).
    - Qui entrano in gioco le LSTM (Long Short-Term Memory), reti ricorrenti che catturano sequenze temporali:
      * L’encoder con LSTM analizza la sequenza di engagement.
      * Il decoder con LSTM ricostruisce la sequenza, forzando il modello a imparare pattern discriminanti (ad es. picchi improvvisi tipici delle fake news).
* **Contenuto + contesto**:
  + **SAF**: combina SAF/S (contenuto) e SAF/A (social) con autoencoder + LSTM multilayer.
    - Una parte della rete (**autoencoder**) lavora sul contenuto testuale.
    - Un’altra parte della rete (**LSTM**) lavora sul pattern temporale delle interazioni sociali.
    - Le due rappresentazioni vengono fuse per classificare la notizia come vera o falsa.

***RISULTATI***

Dalle analisi è risultato che:

* Modelli basati solo sul **contenuto**: prestazioni attorno al 60–65% di accuratezza.
* **SAF/S e SAF/A** da soli migliorano leggermente (~65–67%).
* **SAF (fusion)** hanno i migliori risultati complessivi, con accuratezza fino al 69% su PolitiFact e ~69% su GossipCop, mostrando che la combinazione di contenuto e contesto sociale migliora la detection.

Il contesto sociale (pattern di interazioni, bot, sentiment) risulta fondamentale per distinguere notizie vere e false.

***DATASET***

**FakeNewsNet**, che integra informazioni da più dimensioni:

* **News Content**: articoli etichettati come veri o falsi da fonti di fact-checking affidabili.
  + PolitiFact → ambito politico.
  + GossipCop → gossip e intrattenimento.
* **Social Context**: interazioni degli utenti sui social media (Twitter).
* **Spatiotemporal Information**: localizzazione degli utenti (da profilo o tweet geolocalizzati) e sequenza temporale di diffusione (quando e come gli utenti interagiscono con le notizie).

PAPER 21

**CSI: A Hybrid Deep Model for Fake News Detection**

* Natali Ruchansky, Sungyong Seo, Yan Liu
* 2017
* <https://arxiv.org/abs/1703.06959>

NOTE: utilizza altre informazioni oltre al testo

***METODO***

L’idea chiave: una fake news si può riconoscere osservando tre caratteristiche principali:

* Testo dell’articolo.
* Response: la risposta/propagazione che riceve.
* Source: chi (quali utenti) la diffonde.

Il modello proposto è **CSI**, composto da tre moduli:

* **Capture (testo + risposta)**: Usa un LSTM (RNN) per modellare l’andamento temporale delle interazioni con un articolo (ad es. retweet o post).
* **Score (sorgenti / utenti)**: Mira a catturare il comportamento degli utenti che condividono notizie. Costruisce un grafo implicito utente-utente (basato sugli articoli condivisi in comune).
  + Usa una rete fully connected per assegnare a ciascun utente un embedding e uno score (quanto l’utente appare “sospetto”).
* **Integrate (fusione finale)**: Combina la rappresentazione dell’articolo con gli score degli utenti che l’hanno diffuso. In pratica, Integrate mette insieme il “profilo di propagazione” e il “profilo degli utenti” per decidere.
  + Output finale è la probabilità che un articolo sia fake

***RISULTATI***

CSI viene confrontato con diversi baseline: SVM-TS, DT-Rank, DTC, LSTM-1, GRU-2. CSI ottiene migliori prestazioni su entrambi i dataset.

* Twitter: Accuracy 0.892, F1 0.894 (vs ~0.83 dei migliori modelli concorrenti).
* Weibo: Accuracy 0.953, F1 0.954 (vs ~0.91 dei migliori concorrenti).

CSI ha ~52K parametri, molto meno di modelli RNN più complessi (es. GRU-2: 621K). Richiede meno dati di training per ottenere buone performance.

***DATASET***

Due dataset reali che contengono testo, utenti e interazioni temporali:

* **Twitter**: 233.719 utenti, 992 articoli, ~600.000 interazioni, metà fake e metà vere.
* **Weibo (social cinese)**: 2,8 milioni di utenti, 4.664 articoli, ~3,7 milioni di interazioni, anch’essi bilanciati tra fake e veri.

PAPER 22

**Detecting rumors from microblogs with recurrent neural networks**

* Jing Ma, Wei Gao, Prasenjit Mitra, Sejeong Kwon, Bernard J. Jansen, Kam-Fai Wong, Meeyoung Cha
* 2016
* <https://ink.library.smu.edu.sg/sis_research/4630/>

NOTE: Si basa su Rumors

***METODO***

L’idea è sostituire i metodi basati su feature ingegnerizzate a mano (contenuti, utenti, pattern di diffusione) con un approccio deep learning che sfrutti **Recurrent Neural Networks (RNN)**, capaci di catturare le **dinamiche temporali della diffusione delle informazioni**.

L’idea centrale è che un singolo tweet/post è troppo corto per dire se un evento sia vero o no. Quindi il modello lavora a livello di evento, cioè un insieme di post che parlano della stessa notizia.

* **Raccolta dei post per evento** (es. rumor su un disastro naturale).
* **Suddivisione in intervalli temporali**. Invece di considerare decine di migliaia di post uno per uno, i messaggi vengono raggruppati in “finestre temporali”.
* **Rappresentazione dei testi**. Per ogni intervallo si costruisce un vettore con i termini più rappresentativi (top-5000 parole con tf-idf).
* **Sequenza temporali**. Gli intervalli diventano i passi di una sequenza, che è l’input per la RNN.
* **RNN con unità specializzate** (tanh, LSTM, GRU). Serve a catturare la dinamica dell’informazione nel tempo:

Gli autori sperimentano più varianti di RNN, con complessità crescente:

* **tanh-RNN**
  + modello base con funzione di attivazione tanh;
  + limita la cattura di dipendenze a lungo termine.
* **LSTM-1 e GRU-1**
  + RNN con unità a porte (LSTM e GRU);
  + hanno meccanismi di memoria che permettono di conservare o dimenticare informazioni importanti nei vari passi temporali.
  + aggiunta di un embedding layer (100 dimensioni) per ridurre complessità e trasformare input sparsi in rappresentazioni dense.
* **GRU-2**
  + architettura con due layer GRU (profondità maggiore);
  + il primo cattura le dinamiche locali, il secondo apprende interazioni più complesse e robuste, utile soprattutto su dati rumorosi (es. Twitter).

***RISULTATI***

Viene fatto un confronto con baseline (SVM, Decision Tree, Random Forest, modelli con features ingegnerizzate). RNN outperforma tutti i modelli tradizionali su entrambi i dataset.

* Su Twitter:
  + Miglior baseline (SVM-TS): 80.8% accuracy
  + GRU-2: 88.1% accuracy
* Su Weibo:
  + Miglior baseline (SVM-TS): 85.7% accuracy
  + GRU-2: 91.0% accuracy

RNN (in particolare GRU-2) riesce a riconoscere rumors più velocemente e con più precisione rispetto ai modelli tradizionali e anche rispetto a servizi online di fact-checking.

***DATASET***

* Twitter: Basato su eventi confermati da Snopes.com.
  + 498 rumors, 494 non-rumors.
  + ~1,1 milioni di post da ~491k utenti.
* Weibo: Rumors raccolti dal centro di gestione di contenuti di Sina Weibo.
  + 2.313 rumors, 2.351 non-rumors.
  + ~3,8 milioni di post da ~2,7 milioni di utenti.

PAPER 23

**EANN: Event Adversarial Neural Networks for Multi-Modal Fake News Detection**

* Yaqing Wang, Fenglong Ma, Zhiwei Jin, Ye Yuan, Guangxu Xun, Kishlay Jha, Lu Su, Jing Gao
* 2018
* <https://dl.acm.org/doi/10.1145/3219819.3219903>

***METODO***

Gli autori propongono **EANN (Event Adversarial Neural Network)**, un framework che punta a imparare rappresentazioni invarianti rispetto all’evento, ma discriminative per distinguere fake news da reali.

Il framework è composto da tre componenti principali:

* **Multi-modal Feature Extractor**
  + Estrae feature da testo e immagini (tipici contenuti dei post social).
  + Testo: usano Text-CNN (CNN con filtri di diversa finestra, pooling e fully connected finale).
  + Immagini: usano VGG-19 pre-addestrata, con un fully connected finale per ridurre la dimensionalità.
  + Le rappresentazioni testuali e visive vengono concatenate per formare un embedding multi-modale.
* **Fake News Detector**
  + Una fully connected con softmax che predice se un post è fake o reale.
  + Riceve come input la rappresentazione multi-modale.
  + Ottimizzato tramite cross-entropy loss.
* **Event Discriminator**
  + Una rete feedforward a due layer che prova a predire l’evento di appartenenza del post (tra K eventi noti).
  + La logica è quella del **domain adversarial training**:
    - L’estrattore di feature cerca di foolare il discriminatore, producendo feature invarianti rispetto all’evento.
    - Si gioca quindi una partita minimax tra estrattore e discriminatore.
  + Implementato tramite **gradient reversal layer**: durante il backprop la perdita del discriminatore è invertita per spingere l’estrattore a rendere le feature “event-agnostic”.

L’estrattore produce feature che devono essere utili per distinguere fake da real, ma non devono contenere tracce dell’evento. Così il modello **impara pattern generali delle fake news** (es. incoerenza tra testo e immagine, linguaggio fuorviante, immagini manipolate), piuttosto che keyword legate a un singolo evento.

***RISULTATI***

Nelle analisi vengono considerate alcune baseline:

* **Single-modality**:
  + Text (solo CNN su testo).
  + Vis (solo VGG19 su immagini).
* **Multi-modal**:
  + VQA (Visual Question Answering adattato a binary classification).
  + NeuralTalk (captioning model adattato).
  + att-RNN (stato dell’arte precedente, con meccanismo di attenzione multimodale).
* **EANN-**: variante senza event discriminator.

Sui due dataset sono stati ottenuti questi risultati:

* **Twitter**:
  + Best baseline = att-RNN (Acc 0.664, F1 0.676).
  + EANN- = Acc 0.648, F1 0.617.
  + EANN = Acc 0.715, F1 0.719 → +10.3% accuracy e +16.5% F1 rispetto ad att-RNN.
* **Weibo**:
  + Best baseline = att-RNN (Acc 0.779, F1 0.789).
  + EANN- = Acc 0.795, F1 0.800.
  + EANN = Acc 0.827, F1 0.829 → miglioramento significativo.

L’**uso del discriminatore di eventi** **migliora nettamente le performance**, confermando l’importanza di rimuovere le feature evento-specifiche.

Analisi aggiuntive

Aggiungendo il discriminatore anche ai modelli single-modality (Text+ e Vis+), le performance migliorano. Visualizzazioni t-SNE mostrano che EANN produce feature meglio separabili tra fake e real.

***DATASET***

* **Twitter** (MediaEval):
  + 7.898 fake news, 6.026 real news, 514 immagini.
  + Split in training (development) e test set, senza eventi sovrapposti.
* **Weibo** (Cina, 2012-2016):
  + 4.749 fake news, 4.779 real news, 9.528 immagini.
  + Fake news verificate dal sistema ufficiale di “rumor debunking” di Weibo.

PAPER 24

**DeClarE: Debunking Fake News and False Claims using Evidence-Aware Deep Learning**

* Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, Gerhard Weikum
* 2018
* <https://aclanthology.org/D18-1003/>

NOTE: considerano altre informazioni oltre al testo

***METODO***

Gli autori propongono un modello di deep learning chiamato **DeClarE**, progettato per predire la veridicità di una affermazione/claim, e per motivare la classificazione tramite l’evidenza testuale (spiegabilità).

DeClarE è un modello end-to-end basato su deep learning, con tre componenti chiave:

* **Rappresentazione del claim**: Il claim viene rappresentato attraverso word embeddings (pre-addestrati, tipo GloVe).
* **Rappresentazione delle evidenze:** Gli articoli che parlano del claim vengono usati come “evidenze”. Si utilizza un meccanismo di attention che valuta quali frasi o parole negli articoli siano più rilevanti rispetto al claim.
* **Rappresentazione della sorgente:** Viene considerata anche la credibilità della fonte (sia del claim, sia degli articoli usati come evidenza). Questa informazione aiuta a migliorare le predizioni (alcune fonti sono storicamente più affidabili di altre).

L’idea chiave è che la veridicità di un claim (affermazione) non si giudica solo dal testo del claim stesso, ma anche dagli articoli di supporto (evidence articles), la fonte dell’articolo e la fonte del claim stesso. Il modello combina questi tre elementi in un framework end-to-end basato su deep learning, senza feature ingegnerizzate manualmente.

* **Input**
  + **Claim**: rappresentato come media degli embedding delle sue parole (GloVe).
  + **Articolo**: rappresentato come sequenza di embedding delle parole.
  + **Fonti** (claim source e article source): rappresentate come embedding addestrati che catturano la credibilità nel tempo.
* **Rappresentazione dell’articolo**
  + Il testo di ciascun articolo viene processato con una biLSTM, che cattura le dipendenze contestuali avanti e indietro.
  + Così si ottiene una rappresentazione vettoriale di ogni parola nell’articolo.
* **Claim-specific attention**
  + Per ogni parola dell’articolo, il modello calcola quanto è rilevante rispetto al claim.
  + Questo genera dei pesi di attenzione che permettono di evidenziare frasi e parole “chiave” (ad esempio dati numerici o espressioni di dubbio come “false reports”, “evidence shows”).
  + In questo modo il modello diventa interpretabile: mostra quali parole negli articoli hanno portato alla decisione.
* **Aggregazione**
  + Si costruisce una rappresentazione “attenzionata” dell’articolo, ossia una media pesata delle hidden states della biLSTM, usando i pesi di attenzione.
  + Questa rappresentazione viene concatenata con:
    - l’embedding della fonte dell’articolo,
    - l’embedding della fonte del claim.
    - Il tutto passa in più layer fully connected.
* **Output per singolo articolo**
  + Dal layer finale si ottiene uno score di credibilità del claim in base a quell’articolo (classificazione binaria o regressione).
* **Aggregazione multi-articolo**
  + Ogni claim è di solito discusso in più articoli. Il modello produce uno score per ciascun articolo.
  + Lo score finale del claim è la media degli score sui vari articoli.

***RISULTATI***

**DeClarE supera i baseline** (come SVM o modelli neural standard senza attenzione).

* L’uso combinato di claim + evidenza + fonte porta ai migliori risultati.
* L’attention permette di mostrare quali parti degli articoli sono state decisive → aumenta la trasparenza.
* In alcuni dataset, il guadagno di DeClarE rispetto ai metodi tradizionali è significativo (anche 5–10 punti percentuali di accuracy).

***DATASET***

Gli autori valutano DeClarE su più dataset, fra cui:

* **Snopes**: dataset con claim annotati come veri/falsi/mezzi veri.
* **Politifact**: fact-checking su affermazioni di politici.
* **NewsTrust**: valutazioni di articoli di news.
* **FactCheck.org**: altra fonte di fact-checking.

PAPER 25

**Truth of Varying Shades: Analyzing Language in Fake News and Political Fact-Checking**

* Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, Yejin Choi
* 2017
* <https://aclanthology.org/D17-1317/>

***METODO***

Nel paper vengono descritti due task diversi, e quindi due modelli diversi:

* **Classificazione della news reliability** (notizie affidabili vs fake news: satire, hoax, propaganda).
  + Il modello usato è **Maximum Entropy classifier (MaxEnt)** con tf-idf n-gram features (fino a trigrammi).
* **Predizione della veridicità delle affermazioni PolitiFact** (su scala 6-class o binaria true/false).
  + Modelli provati per il task:
    - Naive Bayes
    - Maximum Entropy
    - LSTM (con embeddings GloVe + hidden state 300-dim).

***RISULTATI***

Sul modello MaxEnt è risultato:

* F1-score: 0.65 su test set out-of-domain (contro 0.26 random).
* Caratteristiche rilevanti:
  + Trusted → riferimenti concreti (luoghi, date).
  + Satire → parole come “reportedly”, “confirmed” (tono facetious).
  + Hoax → temi politici divisivi (“liberals”, “Trump”).
  + Propaganda → parole astratte (“truth”, “freedom”), temi specifici (vaccini, Siria).

***DATASET***

* **Corpus** **di** **news**:
  + Trusted news: English Gigaword corpus (14k articoli).
  + Satire: The Onion, Borowitz Report, Clickhole.
  + Hoax: American News, DC Gazette.
  + Propaganda: Natural News, Activist Report.
* **PolitiFact** **dataset**:
  + 10,483 affermazioni fact-checkate, 4,366 selezionate come citazioni dirette.
  + Etichettate su scala a 6 classi: True, Mostly True, Half True, Mostly False, False, Pants-on-fire.

PAPER 26

**dEFEND: Explainable Fake News Detection**

* Kai Shu, Limeng Cui, Suhang Wang, Dongwon Lee, Huan Liu
* 2019
* <https://dl.acm.org/doi/abs/10.1145/3292500.3330935>

NOTE: si concentra anche sui commenti degli utenti

***METODO***

Il lavoro non si limita a rilevare fake news, ma introduce spiegabilità: cioè indicare quali frasi dell’articolo e quali commenti degli utenti supportano la classificazione come fake. Il modello proposto è **dEFEND**, e il framework ha quattro componenti principali

* **News Content Encoder**: **Gerarchico**, basato su word encoder e sentence encoder con attenzione. Usa **bi-GRU** per catturare contesto e pesi di attenzione per dare importanza a parole/sentenze più indicative.
  + La notizia viene divisa in frasi, poi in parole.
  + Ogni parola viene trasformata in embedding (inizializzato con GloVe).
  + **Word encoder**: Bi-GRU che legge la sequenza e genera un vettore per ciascuna parola.
    - Meccanismo di attenzione a livello di parola → assegna più peso a parole indicative (es. “scandalo”, “shock”).
  + **Sentence encoder**: Bi-GRU che prende i vettori delle frasi e produce rappresentazioni di frase.
    - Ancora un livello di attenzione a livello di frase → evidenzia frasi più informative (check-worthy).
  + **Output**: un insieme di rappresentazioni delle frasi della notizia.
* **User Comment Encoder: RNN bi-GRU** con attenzione a livello di parola. Estrae rappresentazioni latenti dei commenti, dando più peso a quelli rilevanti.
  + Ogni commento viene convertito in una sequenza di parole, codificate con Bi-GRU.
  + Attenzione a livello di parola. Si evidenziano termini chiave nei commenti (es. “fake”, “dove l’ha detto?”, “non vero”).
  + Output: rappresentazioni dei commenti.
* **Sentence-Comment Co-Attention**: Parte cruciale per la spiegabilità. **Mappa congiuntamente frasi e commenti**, assegnando pesi più alti a quelli semanticamente correlati. Genera un ranking delle frasi check-worthy e dei commenti explainable.
  + Non tutte le frasi e non tutti i commenti sono utili. Alcuni commenti sono rumore (“Grande notizia!”, “LOL”).
  + Si costruisce una **matrice di affinità tra frasi e commenti**, che misura quanto una frase e un commento sono semanticamente collegati. Con questa matrice si calcolano due mappe di attenzione:
    - Attenzione sulle frasi guidata dai commenti.
    - Attenzione sui commenti guidata dalle frasi.
  + In questo modo il modello mette in evidenza le frasi e i commenti che si spiegano a vicenda e che sono più utili per capire se la notizia è vera o falsa.
  + Risultato: **due vettori riassuntivi** (uno per le frasi e uno per i commenti), pesati secondo la loro rilevanza.
* **Fake News Prediction Layer**: Combina le rappresentazioni pesate (contenuto + commenti) e fa la classificazione binaria (fake / real).
  + I due vettori (contenuto + commenti) vengono concatenati.
  + Passano attraverso un classificatore (softmax).
  + Output: probabilità che la notizia sia vera o fake.

***RISULTATI***

Gli esperimenti si sono concentrati su tre domande (EQ1–EQ3):

* **Migliora la performance di classificazione usando contenuti + commenti?**
  + Sì. dEFEND supera 7 metodi SOTA (RST, LIWC, text-CNN, HAN, TCNN-URG, HPA-BLSTM, CSI) in Accuracy e F1.
  + Es. su PolitiFact: F1 = 0.928 (vs ~0.87 migliori baseline).
  + Su GossipCop: F1 = 0.755 (vs ~0.68 migliori baseline).
* **Qual è il contributo dei contenuti e dei commenti?**
  + Entrambi contribuiscono, e la co-attention migliora ulteriormente.
  + Rimuovere commenti o contenuti porta a un calo significativo di F1 (fino a -18% su GossipCop).
* **dEFEND seleziona davvero frasi e commenti spiegabili?**
  + Per le frasi: confrontato con ClaimBuster. dEFEND è meglio di HAN nel trovare frasi check-worthy.
  + Per i commenti: valutazioni su Amazon Mechanical Turk mostrano che i top-5 commenti selezionati da dEFEND sono preferiti dai valutatori rispetto a HPA-BLSTM (NDCG ↑ 28%, Precision@5 ↑ 30%).

dEFEND unisce contenuti della notizia + commenti social con un meccanismo di co-attention. Questo migliora la rilevazione delle fake news e fornisce spiegazioni interpretabili.

***DATASET***

Gli autori utilizzano FakeNewsNet, con due sottodataset

* **PolitiFact**: notizie politiche verificate.
  + 415 news (270 fake, 145 reali), ~90k commenti, ~68k utenti.
* **GossipCop**: notizie di gossip.
  + 5816 news (2230 fake, 3586 reali), ~231k commenti, ~156k utenti.

PAPER 27

**FakeBERT: Fake news detection in social media with a BERT-based deep learning approach**

* Rohit Kumar Kaliyar, Anurag Goswami, Pratik Narang
* 2021
* <https://dl.acm.org/doi/10.1007/s11042-020-10183-2>

***METODO***

Gli autori propongono un nuovo modello, **FakeBERT**, che combina **BERT** (Bidirectional Encoder Representations from Transformers) con **reti convoluzionali (CNN)** per migliorare l’accuratezza nella classificazione delle notizie come vere o false.

Gli autori hanno confrontato diversi approcci:

* **Modelli di Machine Learning classici**
  + Naive Bayes, Random Forest, Decision Tree, KNN.
  + Utilizzavano GloVe o BERT come embedding iniziali.
* **Modelli di Deep Learning standard**
  + CNN con embedding pre-addestrati (GloVe o BERT).
  + LSTM con embedding (GloVe o BERT).
* **Proposta del paper: FakeBERT**
  + FakeBERT è un **ibrido tra BERT e CNN**.
    - BERT: cattura il significato contestuale delle parole (bidirezionale).
    - CNN parallele: estraggono pattern locali (n-gram) con kernel diversi per intercettare diverse strutture linguistiche.
  + Il testo della notizia (titolo + corpo) viene tokenizzato, e viene passato a **BERT-base**. Così, si ottiene un **tensore di embedding contestuali** (ogni parola rappresentata con il contesto).
  + Dopo BERT, gli embedding vanno in **tre CNN 1D parallele**. Ogni blocco CNN è seguito da un max-pooling che riduce dimensionalità e conserva i segnali più forti. Il modello riesce a "vedere" pattern linguistici di diverse lunghezze.
  + Le tre mappe di feature (dai tre blocchi CNN) vengono concatenate, ottenendo una rappresentazione più ricca del testo.
  + Dopo la concatenazione, passano ancora per due layer di CNN + max pooling, per un’ulteriore estrazione di feature astratte.
  + Il risultato viene appiattito (flatten) in un vettore, poi passa attraverso due dense layer, e poi all’output layer

In questo modo, abbiamo BERT che capisce il significato di una parola nel contesto, mentre le CNN parallele catturano diversi pattern linguistici tipici della scrittura fake (esagerazioni, sequenze sospette, costruzioni ricorrenti).

***RISULTATI***

**FakeBERT** ha raggiunto un’accuratezza del 98,90%, superando di circa +4% i modelli baseline.

FakeBERT sfrutta la potenza di BERT come embedding bidirezionale + CNN parallele per catturare n-gram di diversa lunghezza. È quindi più robusto contro l’ambiguità linguistica.

***DATASET***

**“Fake News” di Kaggle** (usato anche in altri studi), relativo alle elezioni presidenziali USA del 2016.

* Dimensioni: ~20.800 articoli.
* Etichette:
  + True (real news): 10.540 articoli
  + False (fake news): 10.260 articoli

PAPER 28

**Detecting fake news by exploring the consistency of multimodal data**

* Junxiao Xue, Yabo Wang, Yichen Tian, Yafei Li, Lei Shi, Lin Wei
* 2021
* <https://pmc.ncbi.nlm.nih.gov/articles/PMC9759663/>

NOTE: parte del modello serve per le immagini, quindi non considera solo testo

***METODO***

Il modello proposto è **MCNN (Multimodal Consistency Neural Network)**. Questo modello neurale multimodale integra 5 sottoreti:

* **Text Feature Extraction Module:** Estrae le caratteristiche semantiche del testo usando BERT e successivamente un BiGRU per catturare le dipendenze temporali tra parole.
  + **Input**: il testo della notizia.
  + Tecniche usate:
    - **BERT**: cattura il significato contestuale delle parole.
    - **BiGRU**: elabora la sequenza di parole in entrambe le direzioni per modellare meglio le dipendenze temporali e contestuali.
  + **Output**: un vettore di caratteristiche testuali (embedding semantico).
* **Visual Semantic Feature Extraction Module:** Usa ResNet50 (pre-addestrata su ImageNet) per ottenere feature semantiche delle immagini. Applica un meccanismo di attenzione per enfatizzare le parti emotivamente rilevanti dell’immagine. Poi utilizza un BiGRU per allineare meglio le feature visive con quelle testuali.
  + **Input**: l'immagine associata alla notizia.
  + Tecniche usate:
    - **ResNet50** (pre-addestrata su ImageNet): estrae feature visive di alto livello.
    - **Meccanismo di attenzione**: assegna più peso alle parti emotivamente forti dell'immagine (molte fake news usano immagini "forti" o teatrali).
    - **BiGRU**: trasforma le feature visive in una sequenza semantica, per allinearle meglio con il testo.
  + **Output**: un vettore semantico dell'immagine.
* **Visual Tampering Feature Extraction Module**: Si concentra sul **rilevamento di immagini manipolate** (tampering, ricompressione). Applica ELA (Error Level Analysis) sulle immagini e usa ancora ResNet50 per estrarre feature indicative di alterazioni.
  + Tecniche usate:
    - **ELA** **(Error Level Analysis)**: evidenzia tracce di manipolazioni o compressioni multiple.
    - **ResNet50**: applicata sull'immagine ELA per estrarre feature indicative di alterazioni.
  + **Output**: feature di tampering dell'immagine.
* **Similarity Measurement Module**: Calcola direttamente la **similarità tra testo e immagine** tramite cosine similarity dopo aver proiettato entrambi nello stesso spazio semantico. Se c’è bassa coerenza testo-immagine, aumenta la probabilità che sia fake news.
  + Il testo e l'immagine vengono proiettati nello stesso spazio semantico.
  + Si calcola la cosine similarity tra i due vettori (quanto sono coerenti fra loro).
  + Una similarità bassa segnala possibile fake news.
* **Multimodal Fusion Module:** Integra tutte le feature (testo, immagine semantica, immagine tampering, similarità) tramite attenzione. Infine, una classificazione softmax predice se la news è fake (0) o vera (1).
  + Riunisce tutte le informazioni:
    - Feature testuali (BERT+BiGRU)
    - Feature visive semantiche (ResNet50+attenzione+BiGRU)
    - Feature di tampering (ELA+ResNet50)
    - Similarità testo-immagine
  + **Attenzione multimodale**: assegna più peso alle feature più rilevanti.
  + **Softmax classifier**: produce la probabilità che la notizia sia fake (0) o vera (1).

***RISULTATI***

Il modello è stato confrontato con diversi approcci:

* **Single-modal**: LIWC (testo), VGG19 (immagini).
* **Multimodal tradizionali**: att-RNN, EANN, MVAE, SAFE, VQA.
* **BERT+MVNN** (testo con BERT, immagini con MVNN, fusione tardiva).

I risultati ottenuti sono:

* **MCNN ha superato tutti i baseline** su accuratezza, precision, recall e F1 in tutti i dataset.
  + D1: Acc. 96.3% (contro 95.6% del miglior baseline).
  + D2: Acc. 94.7%.
  + D3: Acc. 78.4%.
  + D4: Acc. 88.4%.
* È stato mostrato che ogni modulo migliora le prestazioni:
  + Solo testo: buoni risultati.
  + Aggiunta immagine semantica: migliora.
  + Aggiunta similarità e tampering: ulteriore guadagno.
  + Fusione multimodale completa: massime performance.

***DATASET***

Sono stati usati quattro dataset reali, multimodali (testo + immagini):

* D1: **Kaggle + NYT/Washington Post** (20.015 articoli, 11.941 fake, 8.074 veri).
* D2: **MCG-FakeNews** (dataset cinese, da Weibo, ~38k articoli bilanciati fake/real).
* D3: Twitter dataset da **MediaEval 2016** (tweets con testo e immagini/video).
* D4: **PolitiFact** (notizie politiche USA verificate da esperti).

PAPER 29

**SemEval-2017 Task 8: RumourEval: Determining rumour veracity and support for rumours**

* Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, Arkaitz Zubiaga
* 2017
* <https://arxiv.org/abs/1704.05972>

***METODO***

Paper della challenge RumourEval 2017, dove riportano i modelli sviluppati dai vari team. Sono state proposte due sottotask:

* **Subtask A – SDQC stance classification:** Classificazione dell’orientamento di ciascun tweet in una conversazione verso la voce iniziale (source tweet)
* **Subtask B – Veracity classification:** Predire se la voce iniziale è True / False / Unverified, in due varianti:
* **Closed**: solo contenuto del tweet.
* **Open**: tweet + contesto esterno fornito dagli organizzatori (Wikipedia dump + pagine archiviate dei link).

I modelli utilizzati dai team (**veracity classification**) sono combinazioni di classificatori tradizionali e deep learning leggeri, spesso con feature hand-crafted:

* **NileTMRG** (miglior team nella closed variant): Classificazione tre-way (true/false/unverified). Usavano SVM con feature basate sul contenuto del tweet + distribuzione di stance nei reply (es. % di tweet che supportano o negano).
* **IKM**: CNN per rappresentazione del tweet, con softmax per classificazione.
* **ECNU** (migliore nella open variant): Ensemble supervisionato (più modelli combinati), con molto feature engineering: metadati dei tweet, embeddings multipli, distribuzione SDQC.
* **IITP**: SVM con kernel polinomiale, approccio supervisionato.
* **DFKI-DKT**: Logistic regression con sparse word vectors.

***RISULTATI***

**Closed variant** (solo tweet)

* Migliori punteggi: NileTMRG e IKM (0.536)
* Baseline (classe più frequente) è di 0.571, quindi i sistemi non hanno superato la baseline.

**Open variant** (tweet + Wikipedia + contesto)

* Migliore: ECNU (0.464)
* Ancora performance basse.

***DATASET***

PHEME

* Eventi inclusi: Charlie Hebdo, Ferguson, Germanwings crash, Clinton pneumonia rumour, Marina Joyce “kidnapping”, ecc.
* Annotazione: giornalisti professionisti (per la verità delle voci) + crowdsourcing (per stance SDQC).

PAPER 30

**A survey on fake news and rumour detection techniques**

* Alessandro Bondielli, Francesco Marcellon
* 2019
* <https://www.sciencedirect.com/science/article/abs/pii/S0020025519304372>

***METODO***

Il lavoro raccoglie, organizza e confronta i metodi esistenti per la rilevazione di fake news e rumours.

Il paper organizza i metodi in tre grandi categorie:

* **Machine Learning tradizionale**
  + **SVM**: molto usati, buoni risultati con feature lessicali, sintattiche e contestuali. F1 ≈ 0.84–0.87 in alcuni studi
  + **Decision Trees / Random Forest**: competitive, accuracy ≈ 0.75–0.86
  + **Logistic Regression**: usata per stance detection e rumour veracity, accuracy 0.73–0.75
  + **CRF** e **HMM**: per modellare sequenze e dinamiche temporali nelle conversazioni social, accuracy 0.6–0.75
  + **Ensemble methods**: combinazione di più classificatori, fino ad accuracy ≈ 0.77
* **Deep Learning**
  + **RNN (LSTM, GRU)**: ottimi per sequenze di tweet/post; accuracy fino a 0.91 (rumour detection su Twitter/Weibo)
  + **CNN**: usate sia su testo (word embeddings, paragraph embeddings) che su pattern di diffusione; accuracy fino a 0.93 su Weibo
  + **Modelli ibridi CNN+RNN**: combinano testo + metadati, migliori prestazioni sul dataset LIAR
  + **Multi-task learning con LSTM**: per classificare rumours e stance contemporaneamente, risultati variabili (0.36–0.64 su PHEME)
* **Altri approcci**
  + **Retweet/diffusion patterns**: analisi delle dinamiche di diffusione (es. burst iniziale dei rumours falsi).
  + **Anomaly detection**: PCA/FAMD per distinguere post anomali (F1 ≈ 0.79–0.81 su Weibo)
  + **Crowdsourcing**: sfruttare “likes” o stance degli utenti (accuracy fino a 0.99 su Facebook posts)
  + **Computational fact-checking**: basato su knowledge graphs, web search e matching automatico

***RISULTATI***

Non esiste un “modello definitivo” nel paper, ma la survey mostra che:

* Per **fake news**, i modelli basati su contenuto testuale (CNN, RNN) sono i più efficaci.
* Per **rumours**, è cruciale includere informazioni contestuali (rete sociale, stance degli utenti), quindi funzionano meglio modelli ibridi che combinano contenuto + contesto.

In generale, i modelli deep learning hanno superato le performance dei metodi classici, anche se la mancanza di dataset benchmark rende difficile confrontare i risultati in modo assoluto.

***DATASET***

* **LIAR dataset** (dichiarazioni politiche etichettate con veridicità e metadati da PolitiFact).
* BuzzFeed / Facebook dataset (post da publisher mainstream vs. hyperpartisan, annotati da giornalisti).
* **CREDBANK** (60M tweet, annotati per credibilità tramite crowdsourcing).
* **PHEME dataset** (tweet su eventi reali, annotati da giornalisti come rumour/non rumour, true/false/unverified) .
* **Satirical** sources come The Onion, usati per task di “satire detection”.

PAPER 31

**Fake News Detection via Knowledge-driven Multimodal Graph Convolutional Networks**

* Youze Wang, Shengsheng Qian, Jun Hu, Quan Fang, Changsheng Xu
* 2020
* <https://dl.acm.org/doi/10.1145/3372278.3390713>

***METODO***

L’obiettivo del modello proposto è unire testo, immagini e conoscenza estratta da knowledge graph in un unico framework.

Il modello considera due aspetti fondamentali:

* Relazioni semantiche di lungo raggio tra parole nei post (non sempre vicine nel testo).
* Conoscenze di background (es. sapere che “Donald Trump” = presidente USA) che gli utenti usano naturalmente per giudicare la veridicità.

Il modello Proposto è **KMGCN (Knowledge-driven Multimodal Graph Convolutional Network)**. Il modello rappresenta ogni post come un grafo multimodale e poi usa una Graph Convolutional Network (GCN) per estrarre rappresentazioni semantiche robuste, e le componenti principali sono:

* **Knowledge Distillation**
  + Si parte dal testo del post, e tramite **entity linking** vengono riconosciute entità (es. “Michael Bloomberg”, “Democrat”, “Donald Trump”).
  + Per ogni entità si estraggono concetti da **knowledge graph esterni** (es. YAGO, Probase), come relazioni isA (“Donald Trump → American president, Republican”).
  + Queste informazioni arricchiscono i nodi del grafo con conoscenza di background.
* **Graph Construction**
  + Ogni post diventa un grafo eterogeneo con **tre tipi di nodi**:
    - **Nodi testuali**: parole del post.
    - **Nodi visivi**: oggetti rilevati nelle immagini (es. YOLOv3 riconosce "person", "flag", ecc.).
    - **Nodi concettuali**: concetti da knowledge graph.
  + Gli archi vengono creati usando la **Pointwise Mutual Information (PMI)** tra coppie di parole/concetti, così da catturare correlazioni semantiche globali, non solo locali.
* **Graph Convolutional Network**
  + La GCN prende in input il grafo costruito e calcola **embedding dei nodi** tenendo conto delle connessioni.
  + Due strati di GCN permettono di propagare informazioni lungo il grafo. In questo modo, ogni nodo “impara” dal contesto circostante (es. una parola impara dai concetti e dagli oggetti visivi collegati).
  + Viene applicato un global mean pooling per ottenere un **vettore rappresentativo dell’intero post**.

***RISULTATI***

Vengono confrontate diverse baseline

* **SVM-TS** (SVM con feature temporali).
* **CNN** (rappresentazioni locali del testo).
* **GRU** (sequential model per testi variabili).
* **TextGCN** (solo grafo di parole).
* **EANN** (fusione testo+immagini con adversarial learning).

Dai risultati delle performance (Accuracy / F1), **KMGCN vince nettamente** su entrambi i dataset.

* **WEIBO**: Accuracy 0.8863, F1 0.8834 (contro TextGCN 0.8571 e GRU 0.7927).
* **PHEME**: Accuracy 0.8756, F1 0.8764 (contro GRU 0.8374 e TextGCN 0.8282).

Analisi Ablation (varianti del modello):

* **Senza Knowledge Distillation (NoKD)**: performance peggiora.
* **Senza Visual (NoVisual)**: piccola perdita, ma mostra che le immagini aiutano.
* **Senza entrambe (NoKDVisual)**: simile a TextGCN, ma comunque meglio delle baseline.

***DATASET***

* **PHEME**: 5 eventi di breaking news con post etichettati come real/fake.
* **WEIBO**: raccolta di claim da Sina Weibo con testi, immagini e metadati.

PAPER 32

**Hierarchical Propagation Networks for Fake News Detection: Investigation and Exploitation**

* Kai Shu, Deepak Mahudeswaran, Suhang Wang, Huan Liu
* 2020
* <https://ojs.aaai.org/index.php/ICWSM/article/view/7329>

NOTE: considera la propagazione delle notizie sul web

***METODO***

Il lavoro affronta il problema della rilevazione di fake news sui social media non solo tramite l’analisi del contenuto testuale (che spesso imita quello delle notizie reali), ma sfruttando reti di propagazione gerarchiche delle notizie, cioè:

* **Macro-livello**: diffusione globale tramite tweet e retweet.
* **Micro-livello**: conversazioni e risposte sotto i post.

Il modello proposto è **HPFN (Hierarchical Propagation Network Features)**. Non è un modello neurale, ma un framework di feature engineering. Gli autori costruiscono le Hierarchical Propagation Networks (HPN) e ne estraggono **feature** da **tre prospettive**:

* **Strutturali**
  + Profondità della cascata di retweet/conversazioni.
  + Numero di nodi (utenti coinvolti).
  + Massimo out-degree (utente più influente).
  + Numero di cascata con retweet/commenti.
  + Presenza di bot.
* **Temporali**
  + Tempo medio tra retweet successivi.
  + Durata della vita di una cascata.
  + Quanto rapidamente un utente influente condivide la notizia.
  + Frequenza e velocità delle interazioni nelle conversazioni.
* **Linguistici** (solo micro-livello, cioè nelle risposte/commenti)
  + Sentiment ratio (positivo/negativo).
  + Sentimento medio delle risposte.
  + Sentimento nei thread più lunghi.

L’insieme di queste feature è chiamato HPFN vector.

Una volta costruiti i **vettori HPFN**, gli autori testano diversi algoritmi di **machine learning “classici”**:

* Gaussian Naive Bayes (GNB)
* Decision Tree (DT)
* Logistic Regression (LR)
* Random Forest (RF)

Random Forest è quello che ottiene le prestazioni migliori, ma le differenze non sono enormi

***RISULTATI***

Analisi effettuata

* Le fake news hanno reti più profonde e più dense (catene di retweet più lunghe). Si diffondono più rapidamente ma con vita più breve rispetto alle real news. Le conversazioni (micro-livello) sulle fake news sono più estese e spesso con sentiment più negativo.

Feature importance

* Le feature temporali (es. durata delle conversazioni, velocità di propagazione) sono le più discriminative.
* Poi seguono le strutturali (profondità della rete, out-degree).
* Le linguistiche (sentiment) hanno importanza minore ma complementare.

Gli autori confrontano HPFN con vari baseline:

* **Content-based**: RST (stile retorico), LIWC (lessici psicometrici).
* **Propagation-based classici**: STFN (structural+temporal di Vosoughi et al. 2018).
* **Graph deep learning**: GCNFN (Graph Convolutional Networks).

**HPFN supera tutti i baseline** (anche GCN) su entrambi i dataset.

* PolitiFact: F1 ≈ 0.84–0.87.
* GossipCop: F1 ≈ 0.86–0.87.

Combinare HPFN con feature di contenuto (LIWC, RST) migliora ancora. Funziona bene anche con early detection (entro 12–24h dalla pubblicazione).

***DATASET***

**FakeNewsNet** (Shu et al. 2017), che integra dati da:

* **PolitiFact** (notizie politiche etichettate come vere o false da fact-checker professionisti).
* **GossipCop** (notizie di gossip verificate).

PAPER 33

**Rumor Detection on Social Media: Datasets, Methods and Opportunities**

* Quanzhi Li, Qiong Zhang, Luo Si, Yingchi Liu
* 2019
* <https://arxiv.org/abs/1911.07199>

NOTE: raccoglie metodi diversi, ma quelli solo testo sono i metodi classici

Il lavoro raccoglie, analizza e confronta i metodi esistenti per la rilevazione dei rumors nei social media. I metodi sono organizzati in base alle fonti di informazione usate per **classificare i rumors**:

* **Contenuto (testo e immagini)**
  + Analisi linguistica e stilistica (es. uso di emozioni, punti esclamativi, incertezza).
  + Analisi visuale (immagini/video falsi o manipolati).
  + Algoritmi: SVM, logistic regression, reti neurali (CNN, RNN, LSTM).
* **Informazioni sugli utenti**
  + Credibilità del profilo (anzianità account, follower, attività).
  + Modelli che usano feature manuali (es. Random Forest, SVM) o rappresentazioni apprese con network embedding.
* **Propagazione e network**
  + Analisi delle strutture di diffusione (cascate, grafi).
  + Modelli basati su RNN, LSTM e tree-structured NN per catturare la sequenza e la topologia della diffusione.
  + Approcci graph-based (PageRank-like, heterogeneous networks).
* **Stance detection + rumor detection (multi-task)**
  + L’analisi delle reazioni degli utenti (supporto, negazione, domanda, commento) migliora molto le prestazioni.
  + Esempio: modelli con GRU che imparano sia stance che veridicità.
  + Nei contest SemEval 2017 e 2019, i sistemi vincitori hanno usato stance detection insieme a feature di contenuto, utenti e propagazione.

***RISULTATI***

* I **metodi tradizionali** (SVM, Decision Tree, Logistic Regression) funzionano ma sono meno performanti.
* I **modelli neurali** (RNN, LSTM, CNN, reti multimodali) hanno ottenuto i migliori risultati, soprattutto integrando più fonti di informazione (testo + utenti + propagazione).
* L’uso esplicito dello **stance** ha portato a miglioramenti significativi, al punto che i top-systems nei contest lo hanno sfruttato.
* I metodi basati su **propagation structure** risultano più robusti, ma hanno il limite di richiedere dati che spesso non sono disponibili nelle prime fasi della diffusione del rumor.

***DATASET***

Gli autori elencano un set ampio di dataset (Twitter, Facebook, Reddit, Weibo), con informazioni su: Testo, Utenti, Tempo, Propagazione:

* PHEME e PHEME-R (Twitter, usati in SemEval 2017 e 2019),
* Twitter15 e Twitter16,
* Ma-Twitter e Ma-Weibo,
* BuzzFeedNews, Kaggle PolitiFact, FakeNewsNet,
* Facebook Hoax.

Le etichette possono essere binarie (true/false) o ternarie (true/false/unverified).

PAPER 34

**Toward Automated Fact-Checking: Detecting Check-worthy Factual Claims by ClaimBuster**

* Naeemul Hassan, Fatma Arslan, Chengkai Li, Mark Tremayne
* 2017
* <https://dl.acm.org/doi/abs/10.1145/3097983.3098131>

NOTE: non è proprio fake news detection

***METODO***

**ClaimBuster** è un sistema che punta a automatizzare (almeno in parte) il fact-checking. Il paper si concentra sul componente principale che è il **Claim Spotting**, cioè la capacità di identificare frasi che contengono affermazioni fattuali check-worthy (meritevoli di verifica).

Il task consiste nel **classificare le frasi in tre categorie**:

* **NFS – Non-Factual Sentence** (opinioni, dichiarazioni soggettive, domande).
* **UFS – Unimportant Factual Sentence** (affermazioni fattuali ma banali, es. “Domani è martedì”).
* **CFS – Check-worthy Factual Sentence** (affermazioni di interesse pubblico che i fact-checker tenderebbero a verificare).

L’output è un punteggio di check-worthiness per ogni frase.

Ogni frase è rappresentata con circa 6.600 feature:

* TF-IDF su parole (dopo stemming e stopword removal)
* POS tagging (es. numeri, verbi al passato)
* Named Entity Recognition (persone, organizzazioni, luoghi…)
* Lunghezza della frase
* Sentiment score

***RISULTATI***

Algoritmi provati:

* Naive Bayes (NBC)
* Random Forest (RFC)
* Support Vector Machine (SVM) → scelto come modello migliore.

**SVM con feature lessicali+POS+NER** ha le migliori performance. Migliore accordo per CFS e NFS, più difficile l’UFS (classe ambigua).

***DATASET***

Corpus: **trascrizioni dei dibattiti presidenziali** USA dal 1960 al 2012.

* Totale ≈ 28.000 frasi
* Focus su 20.788 frasi dette dai candidati (almeno 5 parole).

Annotazione manuale: creata tramite una piattaforma online, con training per annotatori (studenti, giornalisti, docenti).

PAPER 35

**Unsupervised Fake News Detection on Social Media: A Generative Approach**

* Shuo Yang, Kai Shu, Suhang Wang, Renjie Gu, Fan Wu, Huan Liu
* 2019
* <https://ojs.aaai.org/index.php/AAAI/article/view/4508>

***METODO***

Metodo non supervisionato per rilevare fake news sui social media, superando i limiti dei metodi supervisionati che richiedono grandi dataset etichettati (costosi e difficili da ottenere). L’idea chiave è sfruttare le **interazioni degli utenti** (tweet, like, retweet, reply) come indicatori impliciti delle loro opinioni rispetto alla veridicità di una notizia, stimando contemporaneamente **l’autenticità della notizia** (vera/falsa), e la **credibilità degli utenti** che interagiscono.

Il modello proposto è **UFD (Unsupervised Fake news Detection framework**). Il framework si basa su un modello probabilistico generativo rappresentato da una rete bayesiana.

Si basa su:

* **Notizie**
  + Ogni notizia ha probabilità a priori di essere vera o falsa, modellata da una distribuzione Beta-Bernoulli.
* **Utenti verificati**
  + Ogni utente verificato ha due parametri di credibilità:
    - **sensibilità** (probabilità di considerare vera una notizia realmente vera),
    - **1-specificità** (probabilità di considerare vera una notizia falsa).
* **Utenti non verificati**
  + Le opinioni dipendono sia dalla verità della notizia che dall’opinione dell’utente verificato a cui hanno reagito.

**Collapsed Gibbs Sampling** (un metodo MCMC) consente di stimare contemporaneamente le variabili latenti (verità della notizia e credibilità degli utenti) senza etichette.

* Burn-in e thinning usati per migliorare la qualità delle stime.
* Output: decisione fake/real per ciascuna notizia + credibilità stimata degli utenti.

***RISULTATI***

Viene fatto il confronto tra UFD e metodi unsupervised noti (basati su verità aggregata):

* Majority Voting
* TruthFinder (Yin et al. 2008)
* LTM (Latent Truth Model, Zhao et al. 2012)
* CRH (Conflict Resolution on Heterogeneous data, Li et al. 2014)

Dai risultati principali si vede:

* LIAR dataset:
  + UFD: 0.759 accuracy, superando il migliore baseline (~0.64) di +18%
  + F1-score superiore sia su notizie vere che false.
* BuzzFeed dataset:
  + UFD: 0.679 accuracy, migliore dei baseline (0.55–0.56).
  + Lieve calo di recall su fake news, ma migliore equilibrio generale.

Quindi incorporare i **second-level engagements** (like/retweet/reply), oltre ai soli tweet, porta un guadagno significativo.

***DATASET***

Sono stati testati due dataset reali:

* **LIAR** (Wang 2017): ~12.8k news statement etichettati da PolitiFact
* **BuzzFeed News** (2016 US election): 1,627 articoli, filtrati a 144 news con dati di engagement su Twitter.