**LISTA PAPER (FAKE NEWS DETECTION + CONTINUAL LEARNING)**

PAPER 1

**Lifelong Evolution: Collaborative Learning between Large and Small Language Models for Continuous Emergent Fake News Detection**

* Ziyi Zhou, Xiaoming Zhang, Litian Zhang, Yibo Zhang, Zhenyu Guan, Chaozhuo Li, Philip S. Yu
* 2025
* <https://www.arxiv.org/abs/2506.04739>

NOTA: Studio fatto sui RUMORS, non proprio sulle FAKE NEWS

***METODO***

**C2EFND** (Continuous Collaborative Emergent Fake News Detection): combina Large Language Models (LLMs) e Small Language Models (SLMs) in un processo collaborativo.

* **Two-Stage Active Learning**
  + **Diversity-based sampling**: seleziona campioni nuovi e diversi rispetto a eventi precedenti, così il modello non impara solo variazioni della stessa cosa, ma copre più scenari possibili.
  + **Uncertainty-based sampling**: utilizza le predizioni congiunte di LLM e SLM per selezionare i dati più incerti da etichettare. Si selezionano i dati su cui i modelli (LLM e SLM) sono più incerti nelle loro predizioni, perché sono quelli che portano più informazione utile una volta etichettati.
* **Continual Learning**
  + **SLM**: viene usato un meccanismo di **rehearsal replay** con una **Memory Bank** (quando impara cose nuove, ripassa anche esempi vecchi per non dimenticarli). La memoria è limitata da una soglia massima e gestita tramite k-means clustering sui vettori semantici per mantenere un set rappresentativo e diversificato (per decidere cosa tenere, si fa clustering).
    - Viene usato **RoBERTa** come modello più piccolo, ottimo per la classificazione binaria (real/fake).
    - SLM è addestrato in **modo incrementale**. Non ricomincia da zero ogni volta, ma aggiorna le sue conoscenze man mano che arrivano nuovi eventi.
    - Per **evitare che dimentichi** gli eventi vecchi, usano due strategie:
      * **Rehearsal replay**: conserva nel “memory bank” alcuni esempi passati e li ripassa insieme ai nuovi.
      * **Knowledge distillation**: il nuovo modello impara anche dalle previsioni della versione precedente (come un insegnante che passa le sue conoscenze all’allievo successivo).
    - Quando SLM riceve un esempio, non vede solo il testo della notizia, ma anche informazioni esterne prese da Wikipedia, e le spiegazioni generate dall’LLM. Quindi prende decisioni con un quadro informativo più ricco.
  + **LLM**: è proposto un **lifelong knowledge editing** (si inseriscono nuove conoscenze senza distruggere quelle vecchie) basato su **architettura** **Mixture-of-Experts (MoE)**, che consente aggiornamenti incrementali con poche modifiche ai pesi, senza dover riaddestrare il modello completo (aggiornare solo pezzi specifici del modello)
    - Viene usato **Llama3-8B** Instruct come grande modello.
    - Un LLM ha tanta conoscenza generale, ma non è aggiornato sugli eventi emergenti (es. ultime notizie di cronaca).
    - Non viene fatto il fine-tuning completo (troppo costoso e lento), ma viene usato il **knowledge editing** con un’architettura a **Mixture-of-Experts (MoE)**. Ogni volta che arriva un nuovo evento, si aggiunge un piccolo modulo che impara solo quell’evento, senza modificare gli altri. Così il modello impara cose nuove senza dimenticare le vecchie.
    - LLM non fa solo classificazione, ma fornisce spiegazioni usando **Chain-of-Thought prompting** (passaggi di ragionamento). Queste spiegazioni vengono poi passate allo SLM per aiutarlo a capire meglio.
    - Grazie all’**in-context learning**, l’LLM può usare esempi passati (dal **memory bank**) come “dimostrazioni” per ragionare meglio sui nuovi casi.
* **Multi-Round Collaborative Learning**:
  + La parte più innovativa è che invece di lavorare separati, LLM e SLM collaborano:
    - Arriva una notizia non etichettata.
    - LLM la valuta (con la sua conoscenza aggiornata e le spiegazioni).
    - SLM la valuta (con i suoi criteri di classificazione).
    - Se danno la stessa risposta, il campione viene considerato “pulito” e aggiunto al training.
    - Se danno risposte diverse, il campione è “ambiguo” e viene mandato a un annotatore umano (**active learning**).
  + In questo modo, i due **modelli si correggono a vicenda**, migliorano insieme, e l’uso di annotazioni manuali si riduce al minimo.
  + I dati vengono iterativamente classificati in clean pool e noisy pool.
    - I **clean** **samples** vengono usati per addestrare SLM e per in-context learning con l’LLM.
    - I **noisy samples** vengono rivalutati nei round successivi.
    - Dopo alcuni cicli, anche i campioni rimanenti vengono assegnati

***RISULTATI***

**Baseline di confronto**:

* Metodi solo SLM (RoBERTa, MVAE, CompNet, FTT).
* Metodi ibridi SLM+LLM (ARG, EFND).
* Metodi multi-dominio (EANN, MDFEND, M3FEND, CANMD).

**C2EFND supera tutti** i baseline su entrambi i dataset:

* Pheme: Accuracy fino a 87.2%, F1 fino a 91.5%.
* Twitter16: Accuracy fino a 94.2%, F1 fino a 94.7%.

Migliora sia la capacità di adattamento agli eventi nuovi che la memoria sugli eventi passati.

L’analisi mostra che:

* Senza knowledge editing degli LLM, performance cala molto.
* Senza distillation e replay negli SLM, il modello dimentica eventi passati.
* L’active learning è cruciale per selezionare esempi davvero informativi.

***DATASET***

**PHEME** (rumours vs non-rumours su Twitter)

* Versione “Rumour Detection and Veracity Classification”: contiene conversazioni tweet annotate per veridicità su 9 eventi reali
  + <https://figshare.com/articles/dataset/PHEME_dataset_for_Rumour_Detection_and_Veracity_Classification/6392078>
* Versione originale del 2016 (“rumours and non-rumours”): basata su 5 eventi (Charlie Hebdo, Ferguson, Germanwings Crash, Ottawa Shooting, Sydney Siege)
  + <https://figshare.com/articles/dataset/PHEME_dataset_of_rumours_and_non-rumours/4010619?file=6453753>
* 5 eventi sequenziali in ordine temporale
  + Ferguson Unrest – 1143 post (859 reali, 254 fake)
  + Ottawa Shooting – 890 post (420 reali, 470 fake)
  + Sydney Siege – 1221 post (699 reali, 512 fake)
  + Charlie Hebdo Shooting – 2079 post (1621 reali, 458 fake)
  + Germanwings Plane Crash – 469 post (238 reali, 231 fake)
  + **Totale**: 5782 post (3837 reali, 1925 fake).

**Twitter16**

* disponibile su Kaggle come parte di un dataset chiamato “Rumor Detection Dataset (Twitter15 and Twitter16)”
  + <https://www.pheme.eu/software-downloads>
* 5 periodi/eventi consolidati in ordine cronologico
  + Sandy – 7428 post (3638 reali, 3790 fake)
  + Boston Attacks – 891 post (726 reali, 165 fake)
  + Syrian boy + Eclipse – 744 post (106 reali, 638 fake)
  + Napal – 1283 post (966 reali, 317 fake)
  + Others – 2129 post (354 reali, 1775 fake)
  + **Totale**: 12.475 post (5790 reali, 6685 fake)

PAPER 2

**Fake News Detection Using Passive-Aggressive Classifier**

* Saloni Gupta, Priyanka Meel
* 2020
* <https://www.researchgate.net/publication/345483581_Fake_News_Detection_Using_Passive-Aggressive_Classifier>

***METODO***

**Passive-Aggressive Classifier (PAC)**: decide dinamicamente se aggiornare i pesi o meno in base all’**errore della predizione**. L’algoritmo **aggiorna i pesi del modello in modo incrementale**, senza dover riaddestrare da zero.

* PAC aggiorna i propri parametri **gradualmente**, esempio dopo esempio. Questo lo rende molto utile quando i dati arrivano in streaming o quando il dataset è molto grande e non si vuole ricalcolare tutto ogni volta.
* **Comportamento “passive-aggressive”**:
  + **Passive**: se un esempio viene classificato correttamente, il modello non cambia. Se una predizione è corretta con un buon margine, il modello rimane passivo.
  + **Aggressive**: se un esempio viene classificato male, il modello si aggiorna immediatamente per correggere l’errore (diventa aggressivo). Se la predizione è errata o troppo vicina alla soglia, il modello aggredisce aggiornando i pesi in modo minimo ma sufficiente per correggere l’errore
* Come le SVM (Support Vector Machines), il PAC cerca di trovare un iperpiano che separi le classi (fake vs real news). Ma invece di calcolarlo tutto insieme (operazione costosa), lo aggiorna in maniera incrementale: ad ogni errore, sposta l’iperpiano quel tanto che basta per correggere la classificazione.

**Analisi nel paper**

* Si inizia con il pre-processing del testo
  + Si parte con un dataset di notizie (vere e false)
  + Si puliscono i testi (rimozione punteggiatura, stopwords, ecc.)
  + Si trasformano in vettori numerici con TF-IDF (Term Frequency – Inverse Document Frequency): misura quanto una parola è importante in un documento rispetto a tutto il corpus
* Addestramento del modello PAC
  + Una volta ottenuti i vettori TF-IDF, il modello PAC viene addestrato a distinguere tra “fake” e “real”.
  + Il training procede iterativamente, aggiornando il modello quando sbaglia.

***RISULTATI***

Addestramento del modello PAC sui dataset di fake/real news e confronto con altri metodi esistenti riportati in letteratura (es. Naive Bayes, CNN/RNN, modelli ibridi).

* Il Passive-Aggressive Classifier ha raggiunto un’accuratezza del 97,5%, superiore agli altri metodi considerati.
* La sperimentazione ha mostrato che PAC è veloce (adatto ad ambienti real-time come social media), efficiente in memoria, e robusto rispetto alla classificazione di testi rumorosi.

***DATASET***

* Non citati i dati specifici, vengono considerati due dataset di notizie classificate come vere o false.
* Le notizie sono state pre-processate e trasformate in feature testuali tramite TfIdfVectorizer, che misura l’importanza delle parole in relazione ai documenti.

PAPER 3

**Continual Learning for Fake News Detection from Social Media**

* Yi Han, Shanika Karunasekera, Christopher Leckie
* 2021
* <https://link.springer.com/chapter/10.1007/978-3-030-86340-1_30>

NOTE: nel modello usano altre informazioni oltre al testo del tweet

***METODO***

Viene adottato un metodo **context-based** basato su **Graph Neural Networks (GNNs)** per distinguere i pattern di propagazione di fake news vs real news. Ogni notizia è rappresentata come grafo di propagazione, dove i nodi sono tweet/utenti e gli archi rappresentano relazioni di retweet.

La GNN impara a riconoscere **pattern topologici e dinamici**: considera una **fake news** se i retweet sono concentrati da pochi nodi influenti, mentre considera una **real news** quando c’è una diffusione più distribuita e organica, con reti meno centralizzate.

Il paper confronta le due tecniche HAN (Hierarchical Attention Network) e dEFEND (Explainable Fake nEws Detection) con la loro proposta del Graph Neural Networks (GNNs):

* **HAN (Hierarchical Attention Network):**
  + Analizza prima le parole in ogni frase. Costruisce una **rappresentazione della frase**.
  + Poi analizza le frasi nel documento. Costruisce la **rappresentazione del documento**.
  + In entrambi i passaggi usa un **meccanismo di attention** per dare più peso alle parole/frasi più rilevanti.
  + Nel paper, serve come **baseline “classica”** che si basa sul contenuto testuale della notizia (titolo, corpo del testo).
  + Funziona bene se lo stile linguistico è simile a quello su cui è stato addestrato, ma peggiora molto se i testi appartengono a un dominio diverso.
* **dEFEND (Explainable Fake nEws Detection)**
  + modello multimodale e spiegabile. Non si limita a dire “questa è fake news”, ma cerca anche di spiegare quali parti del testo e quali segnali di contesto hanno portato a quella decisione
  + dEFEND combina **due tipi di informazione**:
    - **Contenuto della notizia**
      * Titolo e corpo del testo vengono processati con modelli NLP (embedding + reti neurali).
      * L’obiettivo è estrarre feature linguistiche che aiutino a distinguere fake news da vere notizie.
    - **Contesto sociale (commenti/risposte)**
      * Gli utenti reagiscono alle notizie (commenti, tweet, risposte).
      * Questi commenti spesso contengono segnali importanti (contraddizioni, dubbi, accuse, supporto, ecc.
  + **Meccanismo di co-attention** (attenzione incrociata)
    - L’idea chiave è che non tutti i commenti contano allo stesso modo, e neanche tutte le frasi nel testo principale.
    - Con un meccanismo di attenzione incrociata (co-attention), dEFEND impara ad allineare frasi del testo con i commenti più rilevanti. Se un commento mette in dubbio un’affermazione, il modello impara a dare più peso a quella parte del testo.
    - Questo lo rende anche spiegabile: si possono evidenziare quali frasi e quali commenti hanno portato alla classificazione finale.
* **GNN basato sui pattern di propagazione – proposta degli autori**
  + Gli autori introducono un modello alternativo che non guarda al contenuto del testo, ma a come la notizia si diffonde su Twitter. Fake news e real news hanno pattern di diffusione diversi (chi le retweeta, in che tempi, da quali tipi di account, ecc.)
  + **Diffusione**: ogni notizia diventa un grafo di cascata:
    - **Nodi** = tweet (incluso l’utente che ha fatto quel tweet/retweet).
    - **Archi** = relazione di diffusione (es. chi retweeta chi, menzioni, temporalità).
    - In questo modo, la diffusione di una notizia non è più solo testo, ma una struttura di grafo.
  + **Algoritmo scelto: DiffPool (Differentiable Pooling)**
    - tecnica di Graph Neural Network (GNN) usata per fare classificazione di grafi interi, non solo dei singoli nodi.
    - qui l’obiettivo è dire se “questo grafo di diffusione” corrisponde a una fake news o a una notizia vera.
  + **Feature dei nodi (gli utenti/tweet)**
    - Non si usano parole, ma dati facili da estrarre via API:Utente verificato o no**,** Età dell’account**,** Numero di follower, amici, liste, preferiti**,** Numero di tweet pubblicati**,** Timestamp del tweet (quanto tempo dopo la notizia iniziale)

**Tecniche di Continual Learning**

* **EWC (Elastic Weight Consolidation)**
  + il modello “ricorda” quali pesi erano importanti per il vecchio task.
  + quando impara il nuovo, aggiunge una penalità se quei pesi cambiano troppo.
  + così cerca di bilanciare l’apprendimento tra vecchio e nuovo dataset.
* **GEM (Gradient Episodic Memory)**
  + si tiene in memoria un piccolo set di esempi dal vecchio dataset.
  + quando si aggiorna il modello sul nuovo dataset, si forza la condizione che la performance su quei campioni non peggiori.
  + quindi il modello impara il nuovo senza “dimenticare” il vecchio.
* **Ottimizzazione di GEM**
  + Nel GEM originale, gli esempi del vecchio dataset vengono scelti a caso. Gli autori mostrano che la scelta degli esempi è fondamentale e propongono metodi più intelligenti:
    - **Maximum Entropy Sampling (MES)**: Preferisce esempi che portano più informazione (alta entropia). In pratica, si scelgono campioni che rappresentano meglio la diversità del dataset.
    - **Support Samples**: Ispirato agli SVM, usa solo i campioni “vicini alla frontiera decisionale” (cioè quelli che aiutano di più il classificatore a distinguere vero/falso). Evita di scegliere esempi troppo facili o troppo difficili.
    - **Strategia Stochastic Greedy**: Una combinazione dei due precedenti, prende campioni con margine medio (né troppo sicuri né errati), e massimizza anche l’entropia in modo “greedy”.

***RISULTATI***

Il modello GNN del paper basato sulla propagazione ottiene prestazioni competitive rispetto agli approcci basati su contenuto.

* Il problema di catastrophic forgetting è evidente con **incremental training diretto**, mentre **GEM e EWC** ripristinano un equilibrio tra vecchi e nuovi dataset (GEM più efficace, con cali <3% sugli old tasks). L’ottimizzazione della selezione dei campioni in GEM porta ulteriori miglioramenti.

***DATASET***

* **FakeNewsNet:** integra due fonti
  + PolitiFact (notizie politiche).
  + GossipCop (notizie di intrattenimento).
* Entrambi includono: testo, immagini, tweet/retweet e profili utente associati
* Gli autori usano soprattutto la parte sociale (propagazione e metadati utenti)

PAPER 4

**Incremental Learning for Fake News Detection**

* Renato M. Silva, Pedro R. Pires, Tiago A. Almeida
* 2025
* <https://journals-sol.sbc.org.br/index.php/jidm/article/view/2542>

***METODO***

Gli autori hanno scelto modelli di apprendimento incrementale, senza dover ricalcolare tutto da zero:

* **Passive-Aggressive (PA)**: quando sbaglia una classificazione, aggiorna i pesi in modo aggressivo (forte correzione). Se indovina, resta passivo (non cambia nulla). È robusto contro il concept drift, perché si adatta velocemente ai nuovi dati.
  + è un algoritmo online: riceve un esempio alla volta.
    - Se classifica **bene**, rimane **passivo** (non aggiorna i pesi).
    - Se **sbaglia**, diventa **aggressivo** (aggiorna i pesi per correggere l’errore).
  + Nel fake news detection i dati cambiano nel tempo (concept drift). PA si adatta velocemente alle nuove notizie senza dover riaddestrare da zero.
* **Multinomial Naïve Bayes (MNB)**: assume che le parole appaiano indipendentemente l’una dall’altra. Si aggiorna facilmente quando arrivano nuovi esempi (per questo è adatto a scenari incrementali)
  + Modello probabilistico basato sulla **regola di Bayes**. Assume che le parole (o feature) siano indipendenti, e per ogni classe (fake/real) calcola la probabilità che un documento appartenga a quella classe.
  + È molto usato in text classification (spam, sentiment, fake news), ed è facile aggiornare le probabilità quando arrivano nuovi esempi (adatto all’apprendimento incrementale)
* **Perceptron**: aggiorna i pesi solo quando sbaglia. Semplice ma efficace in problemi binari come fake/real news.
  + Modello lineare per la classificazione binaria (fake vs real). Aggiorna i pesi solo se sbaglia la classificazione.
  + È molto semplice, adatto a contesti in cui serve un aggiornamento veloce e incrementale. È usato come baseline in tanti problemi di classificazione.
  + Tuttavia, funziona solo se i dati sono linearmente separabili, e non cattura pattern complessi.
* **Stochastic Gradient Descent (SGD)**: aggiorna i pesi passo dopo passo, con ogni nuovo esempio.
  + può simulare diversi modelli lineari (come SVM o regressione logistica).
  + utile per grandi quantità di dati, perché non richiede batch enormi.
  + tuttavia, richiede un tuning accurato di learning rate e regolarizzazione. È rumoroso (gli aggiornamenti possono oscillare).

**Tipi di feedback simulati**: Per rendere l’esperimento realistico, hanno provato 3 scenari

* **Immediato**: feedback subito dopo ogni predizione (scenario ideale, ma poco realistico).
* **Ritardato**: feedback arriva più tardi, dopo un certo numero di articoli (simula revisione manuale).
* **Incerto**: feedback solo su una parte degli articoli (simula casi reali in cui non sempre si sa la verità di una notizia).

**Strategie di aggiornamento**: due modalità diverse di aggiornamento dei modelli

* **Solo in caso di errore**: Il modello si aggiorna solo quando sbaglia. Più realistico, perché gli utenti normalmente danno feedback solo quando il sistema fa un errore.
* **Sempre (anche se corretto)**: Il modello riceve feedback e si aggiorna comunque, anche se aveva predetto giusto. Scenario ideale, usato come baseline di confronto.

***RISULTATI***

Gli studi basati su modelli offline sopravvalutano i risultati perché ignorano il concept drift, mentre l’apprendimento incrementale è molto più realistico ed efficace per la detection di fake news. Il tipo di feedback non è determinante

* **Offline models**: Mostrano un crollo drastico delle performance nel 2020, specialmente durante eventi con forte concept drift (COVID-19 e Elezioni USA). Quindi i modelli statici sono inadatti per il problema reale.
* **Online models**: Mantengono performance stabili o addirittura migliorano nel tempo. L’aggiornamento incrementale permette di adattarsi a nuovi pattern linguistici. **Passive-Aggressive** e **Perceptron** si sono rivelati i più robusti.
* **Feedback**: Migliori risultati con feedback immediato (scenario ottimistico). Ma anche con feedback ritardato o incerto, la perdita è minima (i modelli riescono comunque ad adattarsi).

***DATASET***

* **NELA-GT-2019**: 1,2 milioni di articoli da 260 fonti, pubblicati tra gennaio e dicembre 2019.
* **NELA-GT-2020**: 1,77 milioni di articoli da 519 fonti, pubblicati tra gennaio e dicembre 2020. Include subset specifici su COVID-19 e elezioni presidenziali USA 2020.

PAPER 5

**FakeIDCA: Fake news detection with incremental deep learning based concept drift adaption**

* Shubha Suryawanshi, Anurag Goswami, Pramod D Patil
* 2023
* <https://www.researchgate.net/publication/373713955_FakeIDCA_Fake_news_detection_with_incremental_deep_learning_based_concept_drift_adaption>

***METODO***

**FakeIDCA**, un **ensemble di reti neurali incrementali** che **impara progressivamente** da flussi di notizie, fa **pruning dei classificatori meno performanti** per migliorare le prestazioni, e rileva il **concept drift** (cambiamento nei pattern delle notizie) in tempo reale, attivando strategie di adattamento.

* **Ensemble di 4 reti neurali**
  + Il modello non usa una sola rete neurale, ma un ensemble di 4 reti.
  + Ogni rete è feed-forward fully connected (dense)
  + La funzione di perdita è la binary cross-entropy, standard per problemi binari.
* **Rilevamento del drift**
  + Viene usato il **Page-Hinkley (PH) test**, che monitora statisticamente le deviazioni nelle distribuzioni.
  + Funzionamento semplificato:
    - Calcola la media delle predizioni corrette nel tempo.
    - Tiene traccia delle deviazioni da questa media.
    - Se la **deviazione supera una soglia λ**, segnala che c’è stato un drift (il modello non è più aggiornato ai nuovi pattern)
* **Performance-based Incremental Ensemble Classifier**
  + L’ensemble è formato da 4 reti neurali con due hidden layers (128 e 64 neuroni) + output layer sigmoid.
  + in caso di drift, si calcola l'errore in tempo reale di ogni modello dell'ensemble, e i modelli con errore alto vengono rimossi (pruning). Viene aggiunta una nuova rete neurale addestrata sugli esempi più recenti.
  + La predizione finale è una combinazione pesata delle reti, con pesi proporzionali alla loro accuratezza recente.

**Pruning e aggiornamento adattivo**

* Una volta che il drift viene rilevato, l’ensemble deve adattarsi:
  + Si calcola l’errore in tempo reale di ciascuna rete neurale.
  + Le reti che hanno prestazioni peggiori vengono rimosse (**pruning**).
  + Viene aggiunta una **nuova rete addestrata solo sui dati più recenti**.

***RISULTATI***

Il paper mostra che i modelli statici non sono adatti in un ambiente dinamico come le fake news, mentre un approccio incrementale con adattamento al drift è efficace. FakeIDCA combina ensemble learning, pruning adattivo e drift detection, garantendo prestazioni eccellenti (quasi 100% di accuratezza) anche in presenza di cambiamenti significativi nei dati:

* Scenario 1 (**stesso periodo, no drift**): Tutti i modelli (statici e incrementali) ottengono alte performance.
* Scenario 2 (**D1 → D2, presenza di drift**):
  + I modelli **statici** (Naive Bayes, SVM, Decision Tree, MLP) crollano: accuratezza tra 36% e 48%.
  + I modelli **incrementali** (Adaptive Random Forest, Oza Bagging, Hoeffding Tree, Passive-Aggressive) resistono meglio (74–96%).
  + **FakeIDCA** è il migliore:
    - 97.90% di accuratezza su Fake and Real News Dataset.
    - 99.76% di accuratezza su Getting Real about Fake News Dataset.

***DATASET***

* **Fake and Real News Dataset** (Kaggle): 40.525 istanze
  + <https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset>
  + Divisi in due sottogruppi: D1 (2015–2016, 18.720 articoli) e D2 (2017–2018, 21.805 articoli).
* **Getting Real about Fake News Dataset** (BS Detector Extension): 36.300 istanze
  + https://www.kaggle.com/datasets/mrisdal/fake-news

PAPER 6

**Real-Time Fake News Detection on the X (Twitter): An Online Machine Learning Approach**

* Jun Liu, Piyush Vyas, Shengjie Xu
* 2024
* <https://aisel.aisnet.org/amcis2024/social_comp/social_comput/15/>

***METODO***

Sono stati confrontati algoritmi di Online Machine Learning (implementati con la libreria Python River):

* **Passive-Aggressive Binary Classifier (PA-BC)**: apprende incrementando/adattando i pesi in caso di errore.
  + Se il modello classifica correttamente, resta “passivo” (non aggiorna i pesi).
  + Se sbaglia, aggiorna “aggressivamente” i pesi per correggere l’errore.
* **Approximate Large Margin Algorithm (ALMA)**: robusto agli errori statistici, riduce bias.
  + È simile a un **classificatore a margine ampio** (tipo SVM), ma semplificato per l’uso online.
  + Cerca di massimizzare il margine tra classi aggiornando i pesi con approssimazioni matematiche efficienti. “**Approximate**” perché non calcola il margine ottimale esatto (troppo costoso), ma usa **approssimazioni efficienti per aggiornare i pesi**.
  + È robusto contro rumore/statistiche imperfette (riduce il rischio di bias nei dati)
* **Multinomial Naïve Bayes (MNB)**: adatto per text classification, sfrutta l’indipendenza condizionale.
  + È un modello probabilistico che assume che le parole di un testo siano indipendenti tra loro dato il tipo di classe (rumor o non-rumor). Considerando ogni parola indipendente dalle altre (dato che sappiamo se il testo è fake o no), calcola la **probabilità che un testo appartenga a una classe** in base alle **frequenze delle parole**.
* **Hoeffding Tree Classifier (HDT)**: decision tree incrementale, efficiente su stream.
  + È una **variante di decision tree** pensata per i data stream. Non costruisce l’albero intero subito, ma **aggiorna i nodi man mano che arrivano i dati**, usando un test statistico (Hoeffding bound) per decidere quando fare split.
  + È efficiente in memoria e veloce. Funziona bene su dati numerici/strutturati, ma meno su testi rumorosi come i tweet.
* **Ensemble** **– AdaBoost**: combina weak learners in modo adattivo, dando più peso ai casi in cui i modelli precedenti hanno sbagliato
  + Addestra un primo “weak learner”, guarda quali esempi ha sbagliato e aumenta il loro peso.
  + Allena un secondo weak learner che si concentra sugli esempi più difficili.
  + Ripete il processo più volte, in modo da **combinare i weak learners** in un forte classificatore.
* **Ensemble – Bagging**: combina tanti classificatori “deboli” (weak learners) addestrati in parallelo
  + Si prendono più campioni casuali (con replacement) dal dataset, e su ciascun campione si allena un classificatore.
  + Alla fine, i risultati vengono aggregati (votazione media/majority voting).

***RISULTATI***

Il migliore è risultato il **Passive-Aggressive Binary Classifier** (PA-BC), con 99.15% di accuratezza. L’OML è più adattabile, leggero in memoria e veloce rispetto a ML/DL batch, perché non richiede retraining completo.

I limiti dello studio riguardano il dataset relativamente piccolo e sbilanciato.

***DATASET***

* PHEME
  + Contenuto: tweet legati ad eventi reali (es. Boston attack).
  + 238 rumor e 3.714 non-rumor.

PAPER 7

**A Continual Learning System with Self Domain Shift Adaptation for Fake News Detection**

* Sebastián Basterrech, Andrzej Kasprzak, Jan Platoš, Michał Woźniak
* 2020
* <https://ieeexplore.ieee.org/abstract/document/10302539>

***METODO***

Il sistema è basato sul feature extraction **Word2Vec** (embedding testuali, dimensione 100), e il **classificatore LSTM** (Long Short-Term Memory).

* **Word2Vec** trasforma le parole in vettori numerici che catturano relazioni semantiche
* In questo lavoro ogni parola viene mappata in un vettore di dimensione 100. Il risultato è che ogni articolo o post viene rappresentato come una sequenza di vettori numerici, che possono poi essere elaborati da una rete neurale.

Il metodo si basa su:

* **Domain shift detection**: identifica quando la **distribuzione dei dati cambia** (usando metriche come Hellinger distance ed Euclidean distance).
  + Per rilevare il domain shift, il sistema misura la differenza statistica tra i dati correnti e quelli visti in passato, usando, **l’Euclidean distance** (distanza geometrica standard) e **l’Hellinger distance** (più robusta in spazi ad alta dimensione).
  + Se la distanza supera una certa soglia, il sistema conclude che è avvenuto un “drift”
* **Model selection**: seleziona il miglior modello già appreso (**Best Matching Model**, BMM) in base alla somiglianza con i nuovi dati.
  + Il sistema mantiene più **modelli già addestrati**, e quando arriva un nuovo blocco di dati, il sistema sceglie il **modello più adatto confrontando i dati correnti** con i “centri” (mass centers) di ciascun modello. Il modello scelto è chiamato Best Matching Model (BMM).
  + Così si evita di ri-addestrare sempre da zero: si riusa il modello più vicino al nuovo dominio.
* **Model update / creation**: se i **dati sono troppo diversi** dai modelli esistenti, viene creato un nuovo modello.
  + Se i dati sono troppo diversi da tutti i modelli in memoria, il sistema crea un **nuovo modello LSTM** addestrato sui dati correnti.
  + Ma la memoria è limitata, quindi i **modelli vengono gestiti in un buffer FIFO** (First In – First Out). Quando il buffer è pieno, il modello più vecchio viene eliminato per fare spazio al nuovo.
  + La decisione di creare un nuovo modello dipende da una seconda soglia (quanto devono essere “diversi” i dati per giustificare un nuovo modello).

***RISULTATI***

I due dataset appartengono a domini diversi (utile per validare la presenza di domain shift), e sono stati fatti:

* **esperimenti offline**: con valutazioni in condizioni stazionarie (senza drift) per avere un “upper bound” di accuratezza
* **esperimenti in streaming**: valutazione della pipeline CL con e senza domain adaptation, e analisi di sensibilità rispetto ai parametri θ1 (drift detection) e θ2 (creazione nuovo modello).

I risultati principali:

* **Con domain shift adaptation**: il sistema mantiene un’accuratezza più stabile nel tempo.
* **Senza adattamento**: si osserva un degrado significativo dell’accuratezza quando i dati cambiano dominio.
* L’approccio proposto supera i baseline di CL senza adattamento.

***DATASET***

* **Kaggle Fake News**: 26.000 articoli (fake e real), provenienti da siti mainstream USA e notizie generate via Mechanical Turk.
* **Constraint@AAAI2021** – COVID19 Fake News Detection: 10.700 post/articoli annotati manualmente (5100 fake, 5600 real) riguardo al COVID-19.

PAPER 8

**Remember Past, Anticipate Future: Learning Continual Multimodal Misinformation Detectors**

* Bing Wang, Ximing Li, Mengzhe Ye, Changchun Li, Bo Fu, Jianfeng Qu, Lin Yuanbo Wu
* 2025
* <https://arxiv.org/abs/2507.05939>

NOTA: considera sia testo che immagini

***METODO***

Il framework si chiama **DaedCmd** (Dynamically Adaptive Experts and Distributions for Continual MMD) e ha tre componenti principali:

* **Base Feature Extractor**: Estrae feature testuali e visive usando **BERT** (per il testo) e **ResNet/ViT** (per le immagini). Le rappresentazioni multimodali vengono fuse con self-attention e contrastive learning.
  + trasforma i dati grezzi (testo + immagine) in rappresentazioni numeriche.
  + Per il **testo** usa **BERT**, un modello pre-addestrato di NLP, per estrarre il significato semantico delle frasi.
  + Per le **immagini** usa **ResNet** o Vision Transformer (ViT), che sono modelli di visione artificiale.
  + Poi **unisce le due rappresentazioni** (testo e immagine) in una feature multimodale unica tramite:
    - **Contrastive learning**: spinge testo e immagine della stessa notizia a essere simili nello spazio delle feature, e diversi da coppie testo-immagine non corrispondenti.
    - **Self-attention**: mescola le due informazioni per catturare relazioni complesse tra testo e immagine.
* **Dynamically Adapted Mixture-of-Experts (MoE)**: Serve a mitigare il catastrophic forgetting. Comprende **expert condivisi** (shared, che catturano conoscenza comune) ed **expert specifici** per evento (isolano parametri per evitare interferenze tra eventi)
  + Per risolvere il problema del catastrophic forgetting. L’idea principale è di non usare un unico blocco di parametri, ma tanti “esperti” (mini-reti specializzate).
    - **Expert condivisi** (shared experts): contengono conoscenza comune utile a tutti gli eventi (ad esempio pattern generali tra testo e immagine).
    - **Expert specifici** **per evento**: si specializzano su un certo evento (ad esempio le fake news legate a un disastro naturale).
  + Come vengono gestiti gli expert:
    - Non si creano a priori, perché gli eventi possono essere tantissimi. Si usa un **Dirichlet Process** (un metodo bayesiano non parametrico) che decide se serve un nuovo expert:
      * Se un evento è **simile** a uno già visto, si usa un expert esistente.
      * Se un evento è molto **diverso**, si crea un nuovo expert.
* **Environmental Dynamics Model**: Modella **l’evoluzione temporale della disinformazione** come **distribuzioni gaussiane**. Utilizza un modello dinamico continuo basato su **Neural ODEs** per predire come cambierà la distribuzione nei dati futuri.
  + per il cambiamento del contesto sociale, perché le fake news non sono statiche, ma cambiano nel tempo (quello che era falso in passato può diventare vero o viceversa).
  + come viene modellato:
    - Si rappresenta l’ambiente come una distribuzione gaussiana delle feature delle fake news.
    - Si usa un modello dinamico continuo (Neural ODE) che impara a prevedere come questa distribuzione evolve nel tempo.
    - Da questa distribuzione predetta si genera una **feature ambientale ê** che viene aggiunta alle altre feature per aiutare la classificazione.

***RISULTATI***

Come baseline e metodi di confronto sono stati usati.

* **6 modelli MMD**: Base (BERT+ResNet), SAFE, MCAN, CAFE, BMR, GAMED.
* **3 metodi di continual learning**: EWC, Replay, LoRAMoE.

Come risultati principali, abbiamo che **DaedCmd** è meglio di tutti i baseline e CL methods in termini di accuratezza e Macro-F1. Miglioramenti particolarmente evidenti su Twitter, il dataset più difficile (accuracy +5% rispetto alla baseline GAMED).

* Rimuovere il **modello dinamico ambientale** (ê) causa il calo più forte (perdita di capacità predittiva su eventi futuri).
* Anche togliere il **Dirichlet process** o **l’expert condiviso** peggiora i risultati, ma meno drasticamente.
* **Analisi del catastrophic forgetting**: DaedCmd riduce molto il degrado delle performance sugli eventi passati (es. su GossipCop, la perdita di accuratezza sul primo evento è ~6.8 contro i ~12 di EWC).

***DATASET***

* **GossipCop**: 12.840 articoli di news (in inglese).
  + https://www.kaggle.com/datasets/akshaynarayananb/gossipcop
* **Weibo**: 9.528 coppie immagine-testo (in cinese).
* **Twitter**: 13.924 post (inglese), con forte squilibrio tra testo e immagini.

**EvolveDetector: Towards an evolving fake news detector for emerging events with continual knowledge accumulation and transfer**

* Yasan Ding, Bin Guo, Yan Liu, Yao Jing, Maolong Yin, Nuo Li, Hao Wang, Zhiwen Yu
* 2025
* <https://www.sciencedirect.com/science/article/abs/pii/S0306457324002371>

NOTE: Nessun pdf

***METODO***

**EvolveDetector** consiste in un **trasferimento di conoscenza** da eventi storici (fake news su eventi già visti) verso nuovi eventi emergenti, senza dover conservare tutti i dati storici e ritirare da zero il training quando appare un nuovo evento

* Hanno progettato un meccanismo di memoria dei modelli a livello di parametri (“**parameter‐level knowledge transfer**”) con:
  + un hard attention‐based knowledge storing mechanism: immagazzinano informazioni su eventi passati tramite una **knowledge memory + event masks**
    - **Knowledge memory**: una “libreria interna” del modello, dove non si salvano direttamente i dati (i tweet o i post), ma le conoscenze parametriche che il modello ha imparato da ciascun evento passato.
    - **Event masks**: sono come etichette o “filtro colorato” applicato alla memoria. Ogni evento ha un suo mask che dice se determinati neuroni/parametri del modello sono stati usati per imparare un determinato evento. Quando arriva un nuovo evento, il modello attiva le parti della memoria che sono simili e le usa per iniziare a imparare più in fretta, invece di ricominciare da zero.
  + quando **arriva un nuovo evento**, EvolveDetector **recupera quelli corrispondenti ad eventi simili** presenti in memoria, per guidare il training sul nuovo evento.
    - quando arriva un nuovo evento, il modello fa un’operazione di retrieval (recupero). Guarda nella memoria quali eventi passati sono più simili al nuovo (tramite embeddings / rappresentazioni vettoriali), e poi riattiva i parametri associati a quegli eventi, così da “trasferire” conoscenza.
  + usano anche **multi‐head self‐attention** per integrare o combinare le feature corrispondenti a questi eventi storici simili, al fine di addestrare un classificatore per il nuovo evento.
    - una volta recuperate queste informazioni dagli eventi simili, bisogna fonderle con le nuove informazioni. per farlo, il modello usa multi-head self-attention (meccanismo tipico dei Transformer). Questo serve a pesare dinamicamente quanto ogni conoscenza passata è utile per spiegare il nuovo evento.
* Sono interessati a “**continual knowledge accumulation and transfer**”, ossia accumulare conoscenza da eventi passati, usarla per i nuovi, ma con un meccanismo leggero che non richiede memorizzazione completa dei dati storici. EvolveDetector **non conserva i dati**, ma **solo parametri** compressi e **maschere**, molto più leggeri.

***RISULTATI***

EvolveDetector ottiene prestazioni migliori nella generalizzazione su eventi nuovi rispetto ai baselines. Lo fa senza necessità di conservare tutti i dati passati, il che lo rende più leggero in termini di storage e più efficiente computazionalmente.

***DATASET***

* Non è chiaro se contengono solo testo o anche immagini oppure altri metadati

**What’s Real News Today? A Multimodal, Continual-Learning Approach for Detecting Fake News Over Time**

* Luca Maiano, Martina Evangelisti, Silvia Bianchini & Aris Anagnostopoulos
* 2025
* <https://link.springer.com/chapter/10.1007/978-3-031-78980-9_24>

NOTE: Nessun pdf

***METODO***

Gli autori propongono di trattare le news come uno stream temporale e sviluppare un approccio di continual learning che apprende man mano che emergono nuovi eventi, mantenendo buone prestazioni sia per i dati nuovi che per i vecchi.

Le due componenti principali sono:

* Un’**architettura multimodale** chiamata **Tri-Encoder** (multimodale + cross-attention), che permette meccanismi di cross-attention tra testo e immagine.
  + l’architettura Tri-Encoder è progettata per gestire l’interazione tra testo e immagini.
  + È un modello end-to-end che prende in input dati testuali e immagini correlate.
  + Fa uso di meccanismi di cross-attention tra testo e immagine, ossia non si limita a una semplice concatenazione delle rappresentazioni unimodali, ma permette che il testo e l’immagine “si parlino” tra loro (via attention).
* Una strategia di **continual learning** per aggiornare il sistema nel tempo, mitigando la perdita delle conoscenze precedenti (forgetting).

***RISULTATI***

Il modello con continual learning raggiunge miglioramenti medi di accuratezza rispetto a una strategia di solo transfer learning.

Migliora anche la F1 rispetto allo scenario “ideale” (cioè addestrare tutto insieme), forse grazie al fatto che apprendere progressivamente aiuta a focalizzarsi su cambiamenti e adattamenti nel tempo.

***DATASET***

* Non è chiaro se contengono solo testo o anche immagini oppure altri metadati